UNIVERSIDADE FEDERAL DE SANTA CATARINA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE PRODUÇÃO

ESTUDO DE UM MÉTODO DE ANÁLISE E AVALIAÇÃO ERGONÔMICA APLICADA AO SISTEMA DE COMERCIALIZAÇÃO DA AGRECO

ROSANGELA FELIPPI

Orientação: Dr. José Luiz Fonseca Da Silva Filho

Co-Orientação: Dr[a]. Edla Maria Faust Ramos

Florianópolis, 2002

UNIVERSIDADE FEDERAL DE SANTA CATARINA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE PRODUÇÃO

ESTUDO DE UM MÉTODO DE ANÁLISE E AVALIAÇÃO ERGONÔMICA APLICADA AO SISTEMA DE COMERCIALIZAÇÃO DA AGRECO

ROSANGELA FELIPPI

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia de Produção, da Universidade Federal de Santa Catarina, como requisito parcial para obtenção do título de Mestre em Engenharia de Produção

Orientação: Dr. José Luiz Fonseca Da Silva Filho

Co-Orientação: Dr[a]. Edla Maria Faust Ramos

Florianópolis, 2002

ROSANGELA FELIPPI

ESTUDO DE UM MÉTODO DE ANÁLISE E AVALIAÇÃO ERGONÔMICA APLICADA AO SISTEMA DE COMERCIALIZAÇÃO DA AGRECO

Esta dissertação foi julgada e aprovada para a obtenção do título de Mestre em Engenharia de Produção, no Programa de Pós-Graduação em Engenharia de Produção da Universidade Federal de Santa Catarina

Florianópolis, 10 de outubro de 2002.

Prof. Edson Pacheco Paladini, Dr. Ph. D.
Coordenador do Curso de Pós-Graduação

BANCA EXAMINADORA

Prof. José Luiz Fonseca da Silva Fº, Dr. Orientador	**Prof. Walter de Abreu Cybis, Dr.**
Profa. Edla Maria Faust Ramos, Drª. Co-Orientadora	**Profa. Ana Regina de Aguiar Dutra, Drª.**

À minha filha, Amabile. Todo meu amor e dedicação.

E à Flora que está nascendo...

AGRADECIMENTOS

Agradeço ao meu orientador, professor Fonseca, pela disponibilidade em ajudar sempre e ter me confiado seus ensinamentos e apoio.

Agradeço em especial à professora Edla, pela acolhida, amizade, ensinamentos e por ter acreditado na minha proposta de trabalho, meus sinceros agradecimentos.

Agradeço à UFSC, e ao PPGEP pela disponibilização de ensinamentos gratuitos de alto nível, seus professores e funcionários.

Ao pessoal da AGRECO, que permitiram minha "intrusão" nos seus afazeres, e tão prestativamente me auxiliaram na tarefa de avaliação

Agradeço à Albertina, pelo apoio integral, amizade, incentivo, palavras de conforto e coragem e pelo Benedito.

Aos membros da banca, por terem aceito o convite de prestigiarem meu trabalho.

Agradeço à toda equipe que trabalha comigo, e souberam conduzir os trabalhos na minha ausência, em especial ao William, que se revelou um ótimo profissional e amigo.

À minha mãe, irmãs que sempre acreditaram na conclusão desse trabalho, e toda a minha família pelo apoio que precisei.

À Patrícia por ter atendido da minha filha, na minha ausência, como se fosse sua filha, não economizando carinho, atenção e amor.

Ao meu marido, pelo incentivo e auxílio.

À minha filha apesar de tão pequena, sempre tão compreensiva e amiga.

Ao Ivo ("Lord Inglês"), sem ele esse trabalho não teria sido concluído na sua totalidade.

Aos demais amigos, colegas de trabalho, pessoas que de alguma forma me auxiliaram nesse projeto, meu muito obrigado.

SUMÁRIO

LISTA DE QUADROS

LISTA DE FIGURAS

LISTA DE SIGLAS E ABREVIATURAS

AGRECO	Associação de Agricultores Ecológicos da Encosta da Serra Geral
BNF	Backus-Naur Form
Ergolist	Ferramenta para avaliar software interativo pela internet
FaSTA	Fast Structured Task Analysis
HCI	Human-Computer Interaction
KRG	Knowledge Representation Grammar
LSC	Laboratório de Sistemas de Conhecimento
LUTAKD	Liverpool University Task Analysis for Knowledge Description
MCT	Memória de Curto Termo
MLT	Memória de Longo Termo
RS	Registro Sensorial
TAG	Task-Action Grammar
TAKD	Task Analysis for Knowledge Description
TDH	Task Description Hierarchy
UFSC	Universidade Federal de Santa Catarina

RESUMO

Avaliações de *softwares* são importantes para verificar se estes estão adequados a critérios e recomendações ergonômicas, e para essa tarefa dispõem-se de muitos métodos. Esta dissertação concentra-se em classificar e avaliar exploratoriamente o método de avaliação ergonômica utilizado por Ramos (1996). Após um estudo e uma síntese dos métodos existentes, na literatura para orientar a classificação do método escolhido, este foi classificado como sendo do tipo analítico empírico-exploratório. A aplicação do método deu-se através de estudo de caso, em que foram aplicados ensaios de interação com usuários reais em um sistema de comercialização da Associação de Agricultores Ecológicos da Encosta da Serra Geral, a AGRECO, na Cidade de Santa Rosa de Lima, interior do Estado de Santa Catarina. Detectou-se que o método proposto encontrava-se pouco detalhado, e foi preciso adotar outros métodos para dar suporte à sua aplicação. Inicialmente, optou-se por uma avaliação heurística para detectar falhas mais aparentes e, em seguida, partiu-se para a preparação dos ensaios de interação que se constituíram fonte de dados para a aplicação. Este estudo teve como resultados: classificação do método; sugestões de inclusões de descrições mais amplas e de novos passos para complementação do método; além de contribuições ergonômicas ao *software*, à medida que este foi sendo aplicado.

Palavras-chaves: análise e avaliação ergonômica de software, métodos de avaliação.

ABSTRACT

Many methodologies of evaluation of software are imporant to check whether the software are suitable to the ergonomic rules. This dissertation aims to classify, apply, evaluate and fit by sample and get the best details of the description of a method of ergonomic evaluation provided by Ramos (1996). A careful study of the available methods in literature was done to show the classification of the chosen method. This study was classified as analytic empiric-exploratory. The application of the method was done through a study-case in which were realized some test of interaction among real users in a commerce system of the AGRECO – Association of Ecological Farmers of Serra Geral – in the city of Rosa de Lima, Santa Catarina State. We've observed that the proposed methodology was not well detailed and it was necessary to adopt other methods to support the first one. At first, we chose an heuristic evaluation to detect possible mistakes then, we prepared tests of interaction that were the source to apply it. From this work we got classification of the method, suggestions to include wider descriptions and new steps to complete the method, besides ergonomic contribution to the software as it was being applied.

Key words: evaluation ergonomic of software and method of evaluation.

1 INTRODUÇÃO

A informática e os sistemas informatizados estão tão presentes em nosso cotidiano que as vezes nem nos damos conta disso, seja através do uso direto de um computador, ou através de sistemas embutidos em aparelhos cada vez mais modernos nos nossos lares, nos eletro-eletrônicos, nos brinquedos, etc. A interação com esses aplicativos dá-se através das interfaces, que devem ser adequadas ao uso a que se destinam. Quanto mais amigável for a interface, melhor ela foi desenvolvida, tendo o que pode ser chamado de qualidade ergonômica, pois o objetivo da interface é facilitar a comunicação entre o homem e o aplicativo, seja ele um visor de um microondas, um painel de uma máquina de refrigerantes, um caixa eletrônico de banco, aplicativos de uso típico de escritórios, como editores de texto e planilhas eletrônicas, ou ainda *softwares* desenvolvidos de forma personalizada para aplicações específicas de empresas.

A ergonomia é um conjunto de conhecimentos científicos relativos ao homem e necessários para a concepção de ferramentas, máquinas e dispositivos que serão utilizados pelo homem, garantindo um máximo de conforto, segurança e eficácia. A prática ergonômica é uma arte que utiliza técnicas e conhecimentos científicos (WISNER, 1987).

A ergonomia possui ainda uma capacidade da interdisciplinar buscando auxílio nas áreas de antropometria, anatomia, fisiologia, neurofisiologia, psicologia cognitiva, sociologia e outras áreas humanas, da saúde e da engenharia. Este conhecimento científico visa a um objetivo prático, que condiciona e justifica a própria existência da ergonomia: a adaptação do trabalho ao homem (SPERANDIO apud MORAES, 2001).

Sobre interações homem-computador, muito se tem pesquisado,

envolvendo diversas áreas do comportamento humano relacionadas ao tratamento de informações, como percepção, raciocínio, e representação mental, que são domínios de estudo da ergonomia cognitiva. O principal papel da ergonomia de *software* é tornar os sistemas cada vez mais adaptados às necessidades humanas. É fundamental agregar esses conhecimentos para gerar interfaces mais amigáveis que facilitem a vida e o trabalho de seus utilizadores, evitando o estresse, as situações de constrangimento e erro. Fica a questão: esses conhecimentos estão sendo utilizados de fato pelos desenvolvedores de software de forma efetiva e contínua? Ou ainda tendem-se a criar *softwares* que privilegiam a lógica de funcionamento e as facilidades de programação ao invés da lógica de utilização?

Em se tratando de *softwares* de aplicações específicas em empresas com os quais se necessita interagir por longos períodos de tempo, estes influenciam diretamente na forma como os indivíduos desempenham suas tarefas. Se os *softwares* forem bem desenvolvidos, privilegiando formas de utilização, dentro de recomendações ergonômicas já definidas, esses produtos levarão o usuário a um ganho de produtividade e satisfação em seu trabalho (CYBIS, 1997).

O fato é que a utilização do *software* adequado, ou pelo menos mais adaptado possível à tarefa e às características de seus usuários, é essencial para o êxito e a satisfação no trabalho. Para diagnosticar *softwares* e suas interfaces, avaliadores utilizam-se de metodologias de avaliação com apoio em recomendações ergonômicas.

Avaliar o *software* é uma tarefa necessária durante o seu ciclo de vida. No processo de concepção (fases iniciais), a avaliação ajuda a identificar parâmetros ou elementos a serem implementados no sistema. Nas fases intermediárias, é útil para validar ou fazer refinamentos do projeto. Na fase final, certifica que o sistema a ser entregue atende aos objetivos pretendidos e às necessidades dos usuários. Caso o sistema já esteja pronto, a avaliação serve para detectar erros e sugerir melhorias para versões futuras. A idéia hoje é construir *softwares* interativos e ergonômicos, e somente se atentar ao fator ergonômico apenas na avaliação de produtos prontos.

Dispõe-se de muitas metodologias para a verificação de *softwares*; nesta dissertação escolheu-se o método de avaliação ergonômica utilizado por Ramos

(1996) que será aplicado em um *software* de comercilização de produtos agroecológicos da Associação de Agricultores Ecológicos da Encosta da Serra Geral (AGRECO), situada na Cidade de Santa Rosa de Lima em Santa Catarina. Esse *software* faz parte de um projeto denominado AgroREDE, que compreende um sistema de planejamento e gerenciamento da produção agroecológica de pequenos produtores, como distribuição da produção, estimativas de vendas, vendas efetivas, controles financeiros, perdas, recebimento de pedidos dos clientes com o objetivo de otimizar a produção e a qualidade dos serviços oferecidos, favorecendo a fixação das famílias em suas pequenas propriedades. Os indivíduos que interagem com o *software* possuem baixa formação escolar e se enquadram na categoria de usuários iniciantes em informática, o que faz ainda mais necessária e importante a aplicação de critérios ergonômicos para elaboração de interfaces mais amigáveis e estimulantes ao uso.

1.1 DEFINIÇÃO DA TEMÁTICA E DO CONTEXTO DA PESQUISA

Esta dissertação é uma pesquisa do tipo exploratória e descritiva, com a aplicação de um estudo de caso experimental. O método utilizado por Ramos (1996) será investigado através de ensaios de interação *in loco,* com a colaboração de usuários reais do sistema. O foco principal deste estudo é a aplicação e a avaliação do método, visando sugerir melhorias no detalhamento das etapas de utilização.

O sistema de comercialização da AGRECO encontrava-se em fase de implantação no momento da aplicação do método, e seus módulos estavam em constantes alterações e ajustes, possibilitando que algumas falhas ergonômicas pudessem ser corrigidas com o auxílio deste trabalho. E aos demais pontos críticos diagnosticados, sugeriram-se recomendações ergonômicas que servirão para melhorias das versões futuras do *software*.

1.2 OBJETIVOS

1.2.1 Objetivo geral

O presente trabalho tem por objetivo geral analisar exploratoriamente o método de avaliação ergonômica utilizado por Ramos (1996) com base num estudo de caso aplicado no sistema de comercialização da AGRECO.

1.2.2 Objetivos específicos

São objetivos específicos deste estudo:

a) aplicar o método utilizado por Ramos (1996) por meio de um estudo de caso;

b) classificar, a partir das determinações oferecidas por vários autores, o método utilizado por Ramos (1996);

c) explicitar e detalhar as etapas do método utilizado por Ramos (1996) inclusive as que venham a ser agregadas na aplicação desta dissertação;

1.3 DESCRIÇÃO DOS CAPÍTULOS

Nesta dissertação são apresentadas as etapas da aplicação e avaliação do método escolhido, enfatizando suas vantagens e limitações, além de serem propostas contribuições para sua melhor descrição.

O Capítulo 2 descreve o método utilizado por Ramos (1996), descrevendo também os modelos de Barthet (1988) e Diaper (1997), que contribuíram para a formulação do método.

O Capítulo 3 sintetiza as abordagens sobre critérios e recomendações ergonômicas disponíveis na bibliografia, indicando a utilização dos critérios e recomendação de Scapin e Bastien (1997), para serem aplicados no método utilizado por Ramos (1996) na etapa de avaliação ergonômica do *software*.

O Capítulo 4 faz uma análise e síntese classificatória dos métodos de avaliação encontrados na revisão bibliográfica, para esclarecer e orientar o processo

de classificação do método escolhido. Neste capítulo serão descritos também técnicas de coleta e registro de dados em ensaios de interação.

O Capítulo 5 demonstra a aplicação do método, por meio de estudo de caso. Durante a aplicação fez-se necessária a inserção de novas etapas ao método, que são as avaliações heurísticas sugeridas por Nielsen (2002), e a preparação dos ensaios de interação recomendados por Cybis (1997).

No Capítulo 6 são feitas conclusões e considerações finais a respeito do estudo e do método.

Nos anexos encontram-se informações relacionadas à aplicação do método em estudo. O Anexo 1 contém a transcrição de uma sessão de interação do estudo de caso. E o Anexo 2 apresenta um exemplo de relatório de avaliação prévia entregue à equipe técnica e desenvolvedores do *software*.

2 MÉTODO UTILIZADO POR RAMOS (1996)

Neste capítulo será apresentado o método utilizado por Ramos (1996), que foi gerado a partir da necessidade da avaliação ergonômica de um sistema de comunicação colaborativa via Internet. O método proposto pela autora aproveita do modelo de Barthet (1989) os conceitos de ergonomia e psicologia cognitiva, buscando orientar o analista na concepção de aplicações computacionais interativas. De Diaper (1989) foi extraída a construção de descrição da análise da tarefa, por ser bastante detalhada e formalizada segundo esta autora.

Para entender melhor o método utilizado por Ramos (1996) é necessário conhecer e descrever melhor os modelos de Barthet (1988) e Diaper (1989), que lhe deram origem. Primeiramente, vamos descrever o modelo de Barthet (1988), e em seguida será explorado o conceito de análise da tarefa, introduzindo assim a descrição do modelo de Diaper (1989), que se constitui de uma sistemática análise de tarefa. E então, serão apresentados detalhadamente o método de Ramos (1996) e sua aplicação no estudo de caso.

2.1 MODELO PROPOSTO POR BARTHET

O modelo de Barthet (1988) possui grande apoio da psicologia cognitiva e da ergonomia. Para a melhoria da comunicação homem-máquina é necessário o máximo de informações a respeito de como o homem trata e processa a informação, particularmente nos diálogos com a interface. Os conceitos expostos abaixo ainda serão melhor esclarecidos no transcorrer desse capítulo:

a) existem diferentes tipos de usuários, que por razões variadas, não utilizam o sistema da mesma maneira;

b) há uma diferença fundamental entre lógica de funcionamento e lógica de utilização;

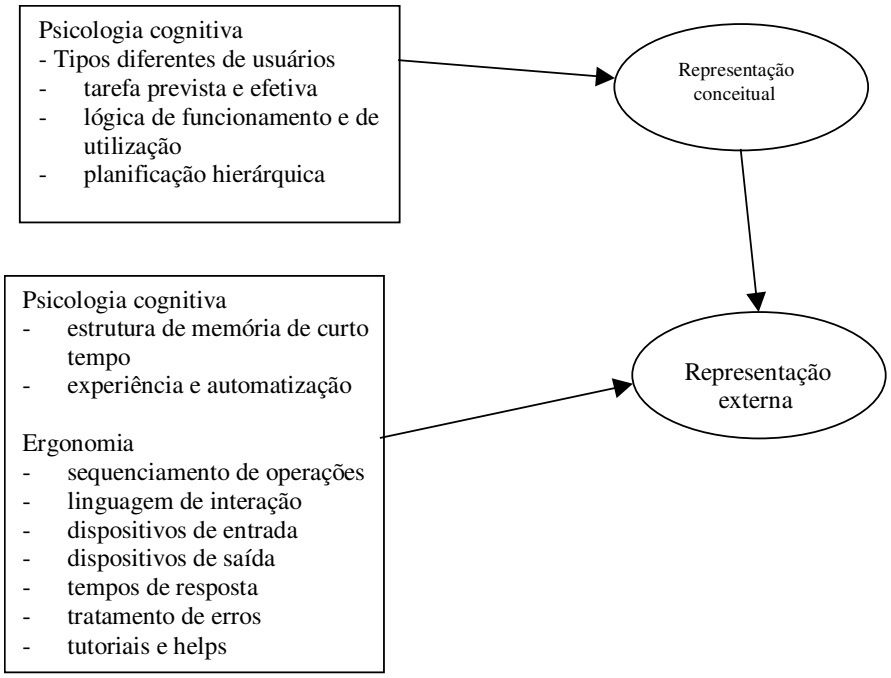
c) a tarefa prevista e a tarefa efetiva são diferentes;

d) o homem organiza uma “planificação hierárquica” para executar uma tarefa;

e) as características de memória de curto termo têm impacto direto na interação homem-máquina.

f) a prática da aprendizagem gera “automatismos”.

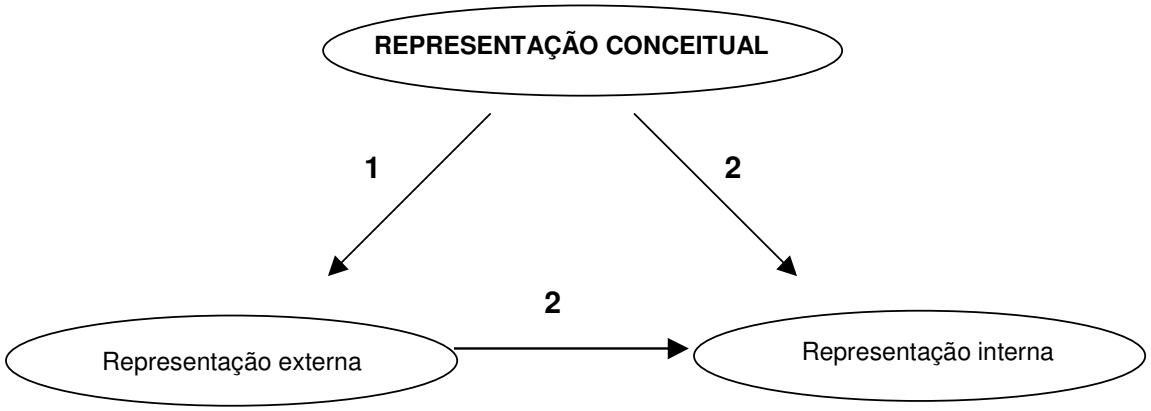
Tais conceitos serão demonstrados através do que Barthet (1996) denomina como representação conceitual e representação externa. A representação conceitual descreve a lógica geral do novo sistema. Nessa representação são incorporados elementos da psicologia cognitiva relacionados com os usuários. A representação externa agrega elementos da psicologia como estruturas de memória e experiências e automatização, descreve a lógica de manipulação do usuário. À representação externa são integrados elementos de ergonomia, que incluem o estudo das condições de trabalho do homem em relação ao uso do computador. Estas condições de trabalho consistem na seqüência de operações, na linguagem de interação, nos dispositivos de entrada, nos dispositivos de saída, nos tempos de resposta, no tratamento de erros e na condução. Essas relações estão representadas no diagrama a seguir.



FONTE: BARTHET, 1988, p. 21

FIGURA 1: INTERAÇÕES ENTRE A REPRESENTAÇÃO CONCEITUAL E A REPRESENTAÇÃO EXTERNA

O modelo proposto por Barthet (1988) para concepção de aplicações interativas integra diferentes pontos de vista: do ergonomista, do usuário e do projetista, que compreendem a construção de três diferentes representações conceituais. A partir da representação conceitual (ponto de vista do analista), deriva-se a representação externa (ponto de vista do usuário, indicada pela seta 1), destas duas, pode-se derivar a representação interna que é elaborada a partir das duas representações anteriores (demonstradas pelas setas de número 2 no diagrama), e identifica o ponto de vista do programador. Estas representações estão demonstradas no diagrama abaixo.



FONTE: BARTHET, 1988, p. 19

FIGURA 2: DIFERENTES PONTOS DE VISTA DAS REPRESENTAÇÕES DE APLICAÇÕES INTERATIVAS

Alguns conceitos são pertinentes para compreender a representação mental do usuário, guiando, desta forma, o analista na concepção de aplicações interativas (BARTHET, 1988). Estes conceitos fazem parte da imagem operativa.

A imagem operativa é a imagem mental que acompanha a ação do trabalho. Quando um usuário é confrontado com um objeto de informação ele é solicitado a agir, constrói uma representação mental do objeto, que sempre é um subconjunto da estrutura integral do objeto, suficiente para o usuário desempenhar determinada ação. Essa representação possui elementos como o laconismo, a deformação funcional, a planificação hierárquica, a diferença entre usuários novatos e experientes, e a noção da tarefa prevista e tarefa efetiva, que serão melhores definidos a seguir.

2.1.1 Laconismo

O usuário cria uma imagem que é apenas um recurso da ação, não absorve nada além do objeto, senão o que é necessário para executar determinada ação.

2.1.2 Deformação funcional

A imagem, nesse caso, criada pelo usuário, é uma réplica deformada do

objeto, constituindo-se de um artifício para acentuar o que é funcionalmente importante para a execução de uma tarefa.

Desse conceito de representação elaborada pelo usuário, Ramos (1996) retira três idéias importantes para a criação de aplicações interativas, que são:

- a partir da noção de laconismo, é possível melhor conceber o vocabulário e a sintaxe dos diálogos, uma vez que para usuários não especializados o entendimento e a solução do problema passam pelo estudo e pela compreensão da linguagem natural, enquanto os especialistas usam vocabulário técnico, especializado, não ambíguo adaptado ao tipo da tarefa;

- O analista de sistemas e o usuário têm uma pequena chance de possuírem a mesma imagem operativa de certa tarefa. O analista tem uma visão ampla e abstrata dos procedimentos para tratamento das informações, já o usuário tem uma imagem local e concentrada na tarefa que deseja executar;

- O processo de decisão e a imagem operativa variam segundo o grau de experiência do usuário, podendo-se prever diferentes representações de uma mesma aplicação para diferentes níveis de experiência de usuários.

2.1.3 Planificação hierárquica

Nesta planificação o usuário elabora um plano de ação para atingir os objetivos. A planificação hierárquica é dinâmica, e representa a estrutura de procedimentos. A imagem operativa, ao contrário, é estática e corresponde às estruturas de dados. A análise desses planos de ação explicita a lógica de utilização que o usuário elaborou, e os subobjetivos que permanecem invariáveis em diversos planos de ação podem ser usados pelo analista para construção da representação conceitual do *software*.

2.1.4 Diferença entre usuários novatos e experientes

Usuários novatos e experientes apresentam comportamentos diferentes, e eles constróem diferentes níveis de imagens operativas.

Para Barthet (1988), a aprendizagem do uso do computador passa pela

diferença entre lógica de funcionamento e lógica de utilização. Alguns usuários aprendem através da lógica de funcionamento, dando o exemplo: “se P então Q”. Outros aprendem através da lógica de utilização, como fazem para atingir seus objetivos, nesse caso: “se o objetivo é Q, então ele faz P”. Para diminuir a diferença entre utilização e funcionamento propõe definir regras de utilização que estejam presentes nas ações possíveis.

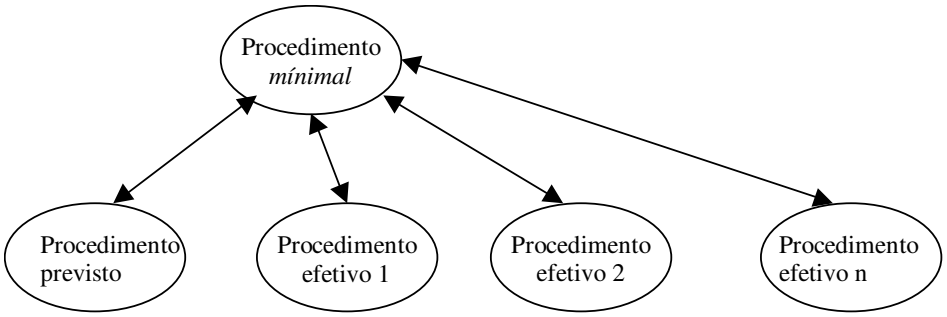
Regras de utilização têm por objetivo definir ações que correspondem mais precisamente para sua realização, passando ao usuário as regras de utilização. O problema é definir boas regras de utilização para ensinar e fazer com que elas correspondam às ações.

2.1.5 Noção de tarefa prevista e tarefa efetiva

A tarefa prevista é aquela idealizada pelo analista que projetou a interface. Ao passo que a tarefa efetiva é a forma como o usuário realmente desempenha a tarefa, que passos ele tomou para alcançar sua meta.

Executar uma tarefa envolve definir um objetivo a ser alcançado ou um ponto a chegar. E o conjunto de passos para alcançá-lo é definido como subobjetivos. Assim, o conjunto de operações que permitem percorrer essas etapas será chamado de procedimentos (BARTHET 1988 apud RAMOS, 1996).

Uma mesma tarefa com um mesmo objetivo pode ser percorrido de maneira diferente por vários usuários. Esses procedimentos diferentes são descritos por Barthet (1988) como procedimentos múltiplos, cujo procedimento previsto corresponde ao procedimento padrão ou recomendado, identificado através de entrevistas. O procedimento efetivo é o que será efetivamente executado pelo usuário, esse procedimento poderá ser checado através de observações. O procedimento *minimal* diz respeito ao mínimo necessário para atingir uma meta, considerando-a como atendida. Pode ser representada pelo diagrama a seguir:



FONTE: BARTHET, 1988 p. 49

FIGURA 3: PROCEDIMENTOS MÚLTIPLOS PARA REALIZAÇÃO DE UMA TAREFA

2.1.5.1 Representação conceitual da aplicação interativa

A representação conceitual é considerada a parte central da modelagem de Barthet (1988, p. 39). Nessa modelagem ocorre a representação do tratamento da informação pelo usuário e sua variabilidade. Na representação conceitual, ainda é necessário diferenciar os parâmetros constantes e os parâmetros variáveis. Os parâmetros variáveis são insumos da análise do trabalho e correspondem ao tratamento do fluxo das informações. Os parâmetros constantes são características gerais do usuário e estão presentes em todas as aplicações interativas.

2.1.5.2 Parâmetros variáveis

São considerados parâmetros variáveis: o posto de trabalho; a descrição do trabalho; a tarefa; os procedimentos que contém a lista de operações; e a lista de pré-requisitos que compreendem a notação de paralelismo. A operação elementar que corresponde à unidade de tratamento da informação que será executada pelo usuário; e a lógica de utilização que é a representação mental do usuário sobre o trabalho.

A decomposição de funções no nível de responsabilidade corresponde ao que é chamado pilotagem (BARTHET, 1988). A pilotagem da aplicação diz respeito à repartição do controle entre o homem e a máquina. Essa divisão oferece ao usuário as noções de controle de execução e a de regulação, o que dá ao usuário a possibilidade de modificar certas condições de execução de maneira a atender ao

objetivo fixado, denominado como latitude decisória.

Existem sistemas onde o controle é feito inteiramente pelo usuário, em que o *software* não impõe nenhuma seqüência de telas ou ações, e não faz nenhum encadeamento automático ou inferência. O usuário é completamente livre para ativar qualquer ordem de seqüências, não havendo controle sobre a coerência entre as atitudes e os objetivos fixados. Já um sistema em que a pilotagem é gerenciada pelo computador garante a coerência entre as ações, mas inibe a criatividade e restringe a flexibilidade nas tomadas de decisão do usuário.

Barthet (1988) propõe, sobre esse problema, a criação de um modelo que explicite o que o usuário deve controlar, e a parte que deverá ser controlada pelo computador. Assim é introduzido o conceito de procedimento *minimal*, segundo o qual para cada tarefa é definido o nível de controle que será dado à máquina, sendo, implicitamente, o restante do controle delegado à decisão do usuário.

A noção de procedimento *minimal* permite definir, na aplicação interativa, a parte que é de controle do computador, e extrair o procedimento mínimo que será executado pelo usuário. O procedimento mínimo possui elementos de natureza diferentes: de um lado, as regras de gestão indispensáveis para assegurar a coerência e a sobrevivência do sistema; e de outro, as regras de comodidade do usuário, chamadas de “bom senso”, que podem ser transgredidas sem pôr em risco o sistema.

As propriedades das operações e dos pré-requisitos, para deixar claro as propriedades de repartição da pilotagem entre o homem e a máquina, estão definidas a seguir.

2.1.5.3 Propriedades das operações

Uma operação é definida como uma seqüência de transações que podem ser executadas para atender a eventos externos ao sistema; as informações necessárias ao desenrolar da operação serão disponibilizadas ao sistema pelo usuário. Elas são descritas pelas entradas, sua natureza, sua condição de disparo/*status*, e sua saída. Assim, define-se que:

a) entradas são uma lista de eventos que podem permitir desencadeamento;

b) a natureza da operação pode ser interativa, automática ou manual;

c) o *status* é ativado ou não, dependendo da realização dos pré-requisitos;

d) as saídas são eventos que podem ser desencadeados por outras operações.

2.1.5.4 Propriedades dos pré-requisitos

As propriedades de pré-requisitos envolvem a noção de sincronismo e pré-condição. O sincronismo é uma proposição lógica (booleana) que, através de operadores lógicos, define as seqüências das operações. Associa-se também a essa propriedade a noção de *delay*, segundo a qual operações dependem da espera de outras operações, ou de datas/tempos para serem disparadas, ou livres a serem executadas a qualquer momento. As propriedades de pré-condições determinam que o evento só poderá ser continuado caso o evento anterior seja considerado, através de valores válidos, como sendo concluído.

2.1.5.5 Propriedades ligadas à pilotagem

A divisão da pilotagem entre o homem e a máquina pode ser obrigatória ou facultativa.

O desencadeamento de uma operação pode ser opcional se depender do disparo do usuário, ou automático, se for disparado pelo computador. A precedência pode ser permanente, se esta existir dentro de todas as descrições de procedimentos da tarefa, sejam elas mínimas, previstas e efetivas. Sua procedência pode também ser indicativa, se ela existir nos procedimentos efetivos e previstos, mas não no procedimento minimal. Desses conceitos construíu-se a figura abaixo para melhor representá-los.

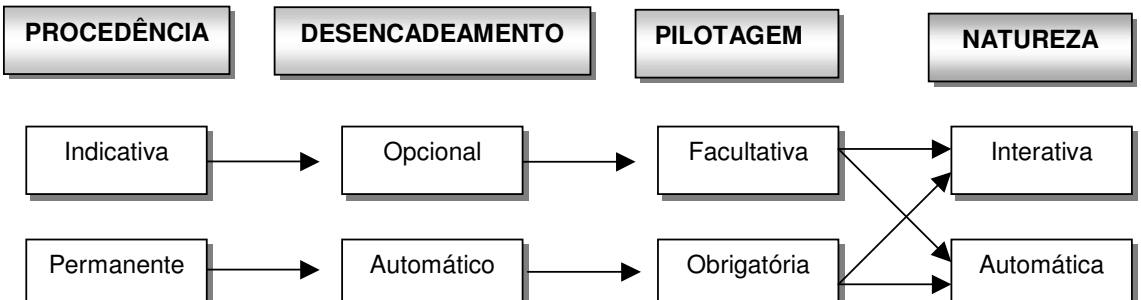


FIGURA 4: DESENCADEAMENTO DE UMA OPERAÇÃO

Uma operação facultativa necessariamente foi disparada através de uma operação opcional; o usuário determinou que ela fosse disparada. A natureza da operação pode ser interativa ou automática e as duas tanto podem ser desencadeadas por operações opcionais ou automáticas.

2.1.5.6 Correspondência entre procedimentos previstos, mínimos e efetivos

> Um procedimento minimal e um procedimento previsto são compatíveis para um mesmo objeto se todas as operações obrigatórias, todas a precedências permanentes e todos os disparos sistemáticos presentes no procedimento previsto forem incluídos no procedimento minimal (BARTHET 1988 apud RAMOS, 1996).

A distinção entre um procedimento previsto e um procedimento efetivo está na ordem metodológica e em eles possuirem a mesma definição de compatibilidade em relação ao procedimento *minimal*.

FONTE: BARTHET apud RAMOS, 1996

FIGURA 5: CORRESPONDÊNCIA ENTRE PROCEDIMENTOS PREVISTOS, MÍNIMOS E EFETIVOS

2.2 MODELO DE DIAPER PARA ANÁLISE DA TAREFA

O modelo de Diaper (1989) é uma forma sistematizada de análise da tarefa. Uma análise da tarefa é uma maneira formal de analisar e reorganizar as características, objetivos e as exigências de um sistema informatizado em questão.

O âmbito da análise da tarefa é bastante amplo. Além da aplicação em tarefas que envolvem diretamente o uso do computador, o analista da tarefa pode modelar dados do mundo real, que não usem computadores diretamente (DIX, 1993). Assim, como análise de sistemas tradicional, a análise de tarefa não é um método limitado a atividades relacionadas ao uso de um computador, embora em análise de sistemas a intenção normalmente é a instalação de um sistema de computador.

Análise da tarefa é o processo de análise que define a forma como as pessoas desempenham seus trabalhos, que coisas eles precisam saber (DIX,1993). Para realizar a análise da tarefa, é preciso saber os passos que devem ser tomados para atingir o objetivo da tarefa, quais os passos que se sobrepõem, quais são os paralelos. Para isso temos a decomposição da tarefa em subtarefas, segundo a ordem na qual elas são executadas.

Para propósitos de produção de materiais de treinamento e documentação, a análise da tarefa é executada sobre sistemas existentes. Porém, a análise da tarefa também pode contribuir para obtenção de requisitos de um sistema inédito, através de observações de uma situação atual.

A análise da tarefa pode ser aplicada para apoiar o projeto de desenvolvimento de *software* em pelo menos três momentos Cybis (1997) e Dix (1993):

a) especificação do sistema;

b) projeto de interfaces;

c) elaboração de manuais e treinamentos.

A maioria das técnicas incluem a classificação e produção de hierarquias, ordenando as entradas através de vários atributos. Isso requer análise subjetivas dos peritos de domínio.

O trabalho de produzir taxonomias não é tarefa trivial. Essa atividade pode

ser feita a mão, usando editores de texto ou ferramentas específicas, que possibilitem recortar determinados níveis, ou realocá-los em outros níveis, facilitando a produção de tais hierarquias (DIX, 1993).

A análise hierárquica da tarefa decompõe uma tarefa em subtarefas. Estes resultados podem ser registrados em forma de texto ou através de diagramas. Técnicas conhecidas constroem taxonomias de objetos durante as tarefas e as ações que são executadas com eles.

Informações para análise da tarefa podem ser obtidas através do estudo de documentação existente, de observação de trabalhadores desempenhando suas tarefas, ou de entrevistas com usuários ou com peritos do domínio. As observações podem ser registradas de diversas formas, desde notas manuscritas até gravação de vídeo e captura por sistemas espiões. A análise, como já foi dito, pode ser utilizada em várias fases do desenvolvimento de sistemas, ou seja, em estágios iniciais, para obtenção de requisitos do sistema novo, ou em estágios finais, como em fases de testes, construção de manuais e documentações. Podem-se ainda obter dados para sistemas inéditos através de análise de sistemas existentes, através de generalizações.

Diaper (2002) afirma: “uma reivindicação é que métodos de análise da tarefa deveriam poder usar e integrar em uma análise, informações derivadas de muitas fontes que usam técnicas de captura diferentes“.

A vantagem do uso da análise da tarefa durante todo o processo de *design* é importante: primeiro para identificar requerimentos do usuário, e só então avaliar decisões de *design*; e, subseqüentemente, para testar protótipos, implantação e entrega do sistema.

Diaper (1992) questiona a ausência de regras claras para uso e aplicação da análise da tarefa em estágios iniciais de desenvolvimento de sistemas. O envolvimento dos usuários e dos desenvolvedores em fases iniciais de desenvolvimetno de sistemas, desencadeia o aumento de qualidade de *software* a ser desenvolvido. O método de analise da tarefa desenvolvido por Diaper (1992) denominado TAKD (Task Analysis for Knowledge Descriptions), é um método detalhado e requer uma forma bastante sistematizada para usá-lo.

O principal papel do método TAKD é analisar os dados oriundos da observação de tarefas relevantes e descrevê-las numa forma representacional única e consistente que especifique o conhecimento que os operadores têm sobre a tarefa que desempenham e os objetos (tecnologia) que usam (DIAPER apud RAMOS, 1996). O TAKD pode ser usado no ciclo de vida de desenvolvimento de sistemas, na produção de especificação de requisitos e com propósito de avaliação.

No centro do método TAKD está a Hierarquia Descritiva da Tarefa (TDH), que possui os seguintes elementos de entrada:

a) lista de objetos específicos;

b) lista de ações específicas.

O método TAKD (DIAPER, 1989) foi inicialmente desenvolvido como parte da aplicação para desenvolvimento nacional de jovens iniciantes em tecnologia de informação. A aplicação de TAKD fazia a análise de dados de observações de tarefas relevantes e as descrevia todas em uma única forma representacional que especificasse o conhecimento geral de informação em tecnologia que aqueles jovens precisavam possuir.

TAKD consiste de um método que gera uma descrição hierárquica de tarefas e uma gramática de representação do conhecimento (KRG). A KRG pode ser transcrita em forma de sentenças. Ela tem sido empregada pelo autor em uma gama de aplicações para vários diferentes propósitos na área de HCI,

O coração da metodologia TAKD é a construção da Task Descriptive Hierarchy (TDH). A razão para a construção de TDH pressupõe que o analista da tarefa faça uma extensão de decisões subjetivas de extensões variadas.

2.2.1 Elementos de entrada de TAKD

TAKD tem como entrada uma ou mais descrições textuais das tarefas provenientes de observações de tarefas. Gerando assim uma lista de ações específicas e uma lista de objetos específicos (DIAPER, 1989), e posteriormente um terceiro tipo que consiste nas seqüências de dados específicos que são similares para ações e objetos específicos.

2.2.2 Objetos específicos

Uma lista de objetos específicos contém todos os objetos que são relevantes para o desempenho da tarefa, ou conjunto de tarefas. A definição de um objeto é algo problemático porque um objeto específico pode ser parte de um objeto maior.

A observação da tarefa produz uma lista de atividades que descreve quais tarefas são desempenhadas. A lista de objetos especificados é simplesmente a lista de cada objeto relevante extraído da lista.

Quando um analista encontra objetos dúbios decide-se por aplicar uma heurística conservativa, a partir da qual redundâncias serão tratadas subseqüentemente, e omissões são consideradas desastrosas.

2.2.3 Ações específicas

As ações específicas, como objetos específicos são primeiro construídas como uma lista em TAKD. As ações específicas são o conhecimento desempenhado por uma pessoa sobre um objeto específico. A heurística conservativa é aplicada na lista de atividades de observações de tarefas. Objetos específicos tendem a ser mais longos que ações específicas. Objetos específicos e ações específicas podem ser descritos simultaneamente. As ações específicas são mais difíceis de identificar, particularmente se a observação da tarefa não envolver gravação em vídeo.

2.2.4 Lista de ações específicas e objetos específicos

A saída do primeiro estágio da análise da TAKD geralmente fornece duas listas, uma de ações específicas e outra de objetos específicos. Entre outros propósitos, isso também ajuda a determinar a freqüência da ocorrência de cada membro da lista, como uma medida de sua importância, e a verificar que algumas ações específicas são impossíveis de serem desempenhadas.

A lista de ações específicas e a de objetos específicos é uma primeira

representação da tarefa, mas, como representação, a lista sozinha é incompleta.

O que foi perdido dos dados da lista de atividades na construção das listas: as informações entre a seqüência das ações; o relacionamento entre especificações; as tarefas desempenhadas; os objetos específicos; a entidade na qual o operador trabalhou.

O próximo passo é a construção da TDH, em que serão descritos os relacionamentos entre ações específicas e seus objetos específicos.

2.2.4.1 Construção da TDH

Talvez o maior propósito em descrever a TDH é tomar uma gama de decisões que o analista precisa tornar explícitas. Isso tem a vantagem de informar ao analista quando cada decisão foi tomada e permitir que diferentes versões de hierarquia, ou parte delas, sejam exploradas. Diaper (1990) relata que nunca conseguiu construir a TDH em um único passo, e teve sempre um número grande de versões, que passaram por várias revisões.

2.2.4.2 Propriedades da TDH

A TDH pode prover uma variedade de diferentes níveis das descrições, as quais poderão ser extraídas para uso subseqüente. Uma TDH é primeiro uma hierarquia, e não uma rede, onde todos os baixos níveis podem ser redescritos como os mais altos, sem o cruzamento ou *loops*. Níveis na hierarquia podem ser omitidos e hierarquias podem ser repetidas. Há três tipos de relacionamentos que podem ser usados mutuamente, exclusivos para cada nível.

a) relacionamento XOR: é a relação mais freqüente, representada no diagrama por uma linha vertical “|” quando um objeto/ação de menor nível puder ser descrito exclusivamente por um ou outro tipo de entidade de maior nível;

b) relacionamento AND: representado no diagrama por uma barra “/”, permite que um objeto/ação possua várias propriedades ao mesmo tempo, é usado apenas nos altos níveis de uma hierarquia;

c) relacionamento OR: representado por uma chave “{ ”, quando pelo menos uma, dentre um conjunto de propriedades, pode ser possuída por um objeto/ação.

Essa representação lógica de TDH pode ser exemplificada por tipos booleanos, onde XOR implica uma e somente uma opção; relacionamento AND, implica todas as opções; e OR é um opcional de AND, com no mínimo uma opção selecionada.

2.2.4.3 Efetivação da TDH

O autor Diaper (1989) aconselha a construção de TDH de forma colaborativa. O trabalho colaborativo torna a construção da TDH mais rápida e fácil de ser feita, pois muitas decisões subjetivas deverão ser tomadas e deverão ser explicitadas. Trabalhando colaborativamente, o analista e demais integrantes irão argumentar sobre seus entendimentos e decidir racionalmente pela melhor escolha.

O primeiro ponto a ser decidido na construção da TDH é o propósito da construção da TAKD. É necessário que o analista esteja bastante familiarizado com a lista de ações e objetos específicos que ele irá descrever. Em seguida deve-se construir uma descrição de muito alto nível para essas ações e objetos, permitindo ao analista usar uma lógica dedutiva e tomar decisões. Essas decisões envolvem alto nível de subjetividade, “pois os nodos de uma hierarquia podem aparecer de muitas formas diferentes e o analista precisa decidir qual é o mais adequado para a análise que ele deseja realizar” (DIAPER apud RAMOS, 1996).

O exercício da construção da TDH, não é efetivado em uma única vez, o processo é ao mesmo tempo indutivo (*bottom-up*) e dedutivo (*top-down*), e muitas vezes seções inteiras de hierarquia devem ser reconstruídas para realocar nodos e níveis. Ramos (1996) atribui uma boa dose de intuição e bom senso, onde o analista deve “sentir” quando chegou ao final da construção da primeira fase da descrição hierárquica da tarefa.

A KRG (Knowledge Representation Grammar) na TAKD fornece a descrição da rota de cada tarefa que usa objetos específicos através da TDH. O produto final desse estágio de TAKD é a lista de sentenças KRG, uma para cada rota através da

hierarquia. É nesse estágio que o analista retorna aos dados das observações originais da tarefa que serão representadas como uma série de passos de tarefas, as quais incluirão ações e objetos específicos, ou descrições que podem ser re-representadas sobre a entidade específica.

Um dos grandes propósitos da TAKD é o da identificação de similaridades gerais entre etapas de tarefas. O processo de generalização envolve a re-representação de etapas da tarefa a um nível de descrição mais geral ou mais alto. Generalização é um processo simples que envolve a remoção de níveis mais baixos da TDH. Alguns desses níveis mais baixos podem ser “cortados” dependendo do propósito da análise.

A intenção das sentenças genéricas da KRG é a de informar ao analista as classes comuns de atividades relacionadas à tarefa e suas freqüências. Essa informação é importante para especificar a funcionalidade de sistemas, enquanto a TAKD é usada durante a prototipação, ou para melhorar um sistema existente; então ela pode ser usada também para avaliação de sistemas. A freqüência de ocorrências de sentenças KRG, comparadas com as necessidades dos usuários, poderá determinar uma reorganização de menus em um sistema, por exemplo.

O crucial aspecto da análise de sentenças KRG é o entendimento de sentenças genéricas de KRG, identificando assim o “conjunto” (macro) comum de tarefas, e não etapas da tarefa.

2.2.5 Sumário do método TAKD:

Abaixo segue a descrição sumarizada do método TAKD, dividida em 11 passos que, segundo Diaper (1989), podem sofrer algumas variações durante as aplicações.

Passo 1: a partir de dados de observações de tarefas, listar todos os objetos e ações específicas (os propósitos de aplicação da análise devem já ter sido estabelecidos);

Passo 2: construir a descrição de alto nível da tarefa, representando-a no formato genérico da KRG;

Passo 3: construir a primeira linhagem da descrição hierárquica da tarefa, usando exemplos de objetos e ações específicas, ou ações genéricas, se apropriado;

Passo 4: construir a descrição hierárquica da tarefa (TDH) final determinando objetos específicos para seus nodos de mais baixo nível;

Passo 5: calcular a freqüência de ocorrência de objetos específicos para todos os nodos da descrição hierárquica da tarefa;

Passo 6: redescrever cada etapa da tarefa observada no formato de sentença da gramática de representação do conhecimento;

Passo 7: gerar sentenças genéricas no formato da gramática de representação do conhecimento, removendo descrições de mais baixo nível depois de alinhar níveis da descrição hierárquica da tarefa, até que o número de sentenças genéricas geradas seja adequado e manejável;

Passo 8: calcular a freqüência das sentenças genéricas de KRG;

Passo 9: interpretar sentenças genéricas de KRG com respeito ao domínio de interesse da tarefa.

Passo 10: repetir o passo 6 com sentenças genéricas de KRG;

Passo 11: extrair repetições de seqüências de sentenças genéricas de KRG e representá-las como um SRG e finalmente um SRG genérico.

O uso completo de TAKD, com a aplicação dos onze passos descritos pelo autor, torna-se inviável sem a utilização de uma ferramenta própria a este fim. Nos anos 80, Diaper (1989) desenvolveu uma ferramenta denominada LUTAKD para auxiliar a aplicação do método em um projeto comercial. Essa ferramenta possuia limitações gráficas em sua interface, mas forçava o analista a seguir rigorasamente os passos definidos no método. A ferramenta foi oferecida para desenvolvimento comercial, porém como o método tinha sido utilizado em um projeto pago, teve problemas de direitos autorais, e após um ano de negociações, quando então o autor retomou os direitos, a empresa que desenvolveria o sistema não tinha mais interesse em desenvolvê-lo, ficando o projeto sem continuação.

A criação da ferramenta LUTAKD demostrou que o método de Diaper ainda

estava pouco desenvolvido. Diaper (2002) aconselha: “se você está desenvolvendo um método então desenvolva ao mesmo tempo uma ferramenta de *software* para suportá-lo, não que isso seja uma base primária para ofertar o método, isso também é importante, mas antes disso, para analisar o método como um produto em si”. Isso também ajuda o projetista do método a identificar o que o método envolve de fato, separar dentro do método que regras são controladas pelo analista e quais são fluxos de representação do método. Essa construção em paralelo força o desenhista do método a considerar onde os analistas têm que aplicar suas habilidades e arte, inteligência e experiências.

A experiência do autor, com a construção de LUTAKD, fez com que essa identificação em si pudesse simplificar o *design* do método, como também aumentou o clareamento da descrição do método. “Se isso tivesse sido feito em 1982, quando foi iniciada a criação do método, é quase certo que o método não teria sido inventado” (DIAPER, 2001, p. 15). Mas como não existia outro método nos anos 80, ele foi uma opção bastante plausível.

Diaper (2002) conclui que restaram três vantagens na utilização do método TAKD.

Primeiro, aquela análise da tarefa em termos de listas de atividade é um modo perfeitamente simples e possível de se fazer uma análise de tarefa.

Segundo, o resultado de uma análise de tarefa deve mapear representações usadas por engenheiros de *software*, pois estes métodos são formais (por exemplo OBJ, Z) ou estruturados (por exemplo, DFDs, ELHs). O autor acredita que é provável que estas áreas devam continuar a se aprimorar.

Terceiro, importante construir representações de hierarquias. Porém os analistas ainda precisam ser ensinados a trabalharem com esses tipos de representações gráficas extremamente comuns.

Após essa demonstração dos modelos de Barthet (1988) e Diaper (1989), pode-se compreender melhor o método utilizado por Ramos (1996). Porém, há uma lacuna quanto à definição do procedimento *minimal* identificada por Ramos (1996) no modelo de Barthet (1988) foi resolvida pela proposta de Diaper (1989), uma vez que Diaper possui uma visão bem detalhada da descrição da análise da tarefa. De

Diaper (1989) Ramos (1996) suprimiu o cálculo das frequências nos nodos da hierarquia de ações e a representação desta hierarquia no formato KRG (Knowledge Representation Grammar). Segundo Diaper (2002), autor de TAKD (Task Analysis Knowledge Description), este passo torna-se demasiadamente trabalhoso e improdutivo sem o auxílio de uma ferramenta informatizada.

2.3 MÉTODO UTILIZADO POR RAMOS (1996)

O método utilizado por Ramos (1996) é uma síntese aprimorada das propostas dos autores Diaper (1989) e Barthet (1989). Segundo Ramos (1996), Diaper (1989) dá conta com consistência e complitude das representações de estruturas taxômicas para as ações e objetos genéricos da tarefa e da subestrutura orientada a metas ou ao plano de executar a tarefa. Porém, a metodologia de Diaper (1989) é omissa com relação à descrição dos procedimentos ou operações constantes na tarefa. Para suprir essa lacuna foi fundamental o apoio conceitual fornecido por Barthet (1989). Por outro lado, pode-se dizer que a metodologia de Diaper (1989) preenche uma ausência bastante importante percebida no modelo proposto por Barthet (1989) no tocante à identificação dos procedimentos mínimos e efetivos, conceitos que são fundamentais naquela abordagem. Tais procedimentos correspondem na proposta metodológica de Diaper (1989) às sequências genéricas e específicas de ações.

Assim sendo, o método utilizado por Ramos (1996) pode ser detalhado através da aplicação dos passos descritos abaixo.

Primeiramente, sugere-se a transcrição de todas as sessões gravadas em fitas com concomitante sinalização de situações problemas; a partir dos dados oriundos da observação da tarefa. Em seguinda, listam-se todos os objetos e ações específicas encontradas. O terceiro passo consiste em construir uma descrição genérica da tarefa. O quarto passo determina a contrução da descrição hierárquica das ações e objetos componentes da tarefa (usando diagramas propostos por DIAPER). O quinto passo compreende a construção da descrição de todas as operações (unidades procedurais ou ações específicas na denominação proposta

por DIAPER) identificadas na tarefa – estas consistiam dos nodos da hierarquia de ações. Tal descrição foi feita tomando como base os modelo de representação conceptual para ferramentas interativas elaborados por Barthet(1988). O sexto passo é definir, segundo o modelo de Barthet, um plano geral de ações para a tarefa, ou seja, obter uma visão macro do sincronismo das ações presentes na tarefa – esta etapa eqüivale à determinação das seqüências de ações que Diaper propõe; e por fim realizar a avaliação ergonômica da aplicação (quando da existência de uma aplicação).

Nessa dissertação procurou-se seguir os passos definidos por Ramos (1996), e à medida que esses passos eram identificados como sendo pouco detalhados, foram feitas inclusões e sugestões de novos passos, que serão detalhados no Capítulo 5, que trata do estudo de caso.

3 CRITÉRIOS E RECOMENDAÇÕES ERGONÔMICAS

A elaboração de interfaces amigáveis, com facilidade de uso, deve-se muito à forma como esses sistemas foram desenvolvidos, quais foram os comprometimentos com o usuário durante o processo de elaboração e desenvolvimento do sistema, e a forma como os desenvolvedores criaram as interfaces, se foi através de facilidades de programação, ou privilegiando a forma de utilização. É a partir desses indícios que se dará a garantia ou não do sucesso de tais sistemas. Para isso, o uso de critérios ergonômicos bem definidos se faz útil tanto nas atividades de concepção como nas de avaliação de sistemas.

Medeiros (1999) faz um estudo minucioso, descrevendo vários conjuntos de critérios ergonômicos, como as oito regras de ouro de Shneiderman (1998), os princípios heurísticos de Nielsen (2002), princípios de projeto centrado no usuário Microsoft, princípios IBM, critérios ergonômicos de Scapin e Bastien (1997) e ISO 9241. Desse estudo, o autor faz diversas conclusões referentes ao grande fator de distinção entre os critérios estudados. Com respeito a Scapin e Bastien (1997), Medeiros (1999, p. 43) conclui:

> (...) reside na diferença entre o grau de detalhamento de cada abordagem. Enquanto a maioria dos trabalhos caracteriza-se pela concisão de suas recomendações, outros – especialmente Scapin e Bastien (1997) – organizam-se de uma maneira coesa e hierárquica. Essa estrutura tende a facilitar a utilização dos requisitos e, por conseqüência, possibilita a obtenção de resultados mais precisos na avaliação e desenvolvimento de interfaces.

Com base no estudo das recomendações e critérios ergonômicos levantados, e levando em conta a conclusão de Medeiros (1999), optou-se por utilizar os critérios ergonômicos de Scapin e Bastien (1997), que serão utilizados

nessa dissertação, merecendo ser descritos em detalhes nesse capítulo. Além dos critérios explorados por Medeiros (1999), encontraram-se ainda princípios de Tognazzini (2001), que embora tenham sido criados para apoiar sistemas baseados em *browsers*, podem ser úteis em avaliações de aplicações informatizadas de maneira geral. Serão também descritas nesse capítulo as Heurísticas de Nielsen (2002), que podem auxiliar no desenvolvimento e avaliação de interfaces interativas, que foram utilizadas na avaliação ergonômica do *software* avaliado nessa dissertação.

3.1 CRITÉRIOS ERGONÔMICOS DE SCAPIN E BASTIEN (1997)

3.1.1 Condução

A condução refere-se aos meios disponíveis para aconselhar, orientar, informar, instruir e guiar o usuário em sua interação com o computador (mensagens, alarmes, rótulos, etc.). Uma boa condução facilita o aprendizado e a utilização do sistema, e reduz a incidência de erros. Esta característica deve permitir ao usuário que ele saiba onde está no sistema, quais possíveis ações pode fazer, e quais suas conseqüências, além de lhe fornecer informações adicionais. Esse critério se subdivide em presteza, agrupamento/distinção de itens, *feedback* imediato e legibilidade.

3.1.1.1 Presteza

A preseteza guia o usuário a realizar determinadas ações, como, por exemplo, a entrada de dados. Engloba todos os mecanismos que permitem ao usuário conhecer as alternativas, em termos de ações, de acordo com o estado ou contexto nos quais ele se encontra. Presteza também significa o *status* da informação sobre o atual estado do sistema bem como informações sobre facilidades de ajuda e modo de acesso.

Uma boa presteza guia o usuário e o poupa de aprender uma série de comandos. Auxilia o usuário a identificar em que parte do diálogo está, e as ações

que tomou para chegar àquele contexto. A presteza ainda facilita a navegação e reduz o número de erros.

3.1.1.2 Agrupamento/distinção de itens

Este critério caracteriza a organização e posicionamento visual das informações, se elas pertencem ou não a determinada classe de itens. Esse item divide-se em agrupamento/distinção por localização e agrupamento/distinção por formato.

3.1.1.2.1 Agrupamento/distinção por localização

O agraupamento/distinção por localização diz respeito à posição e à ordem indicada no relacionamento entre vários itens mostrados na tela. A compreensão da tela pelo usuário depende, dentre outras coisas, da ordem em que os objetos (imagens, textos, comandos, etc.) são apresentados. Os usuários poderão detectar diferentes itens ou grupos de itens e aprender relacionamentos mais facilmente, e a informação estará melhor organizada (por exemplos, em ordem alfabética ou ordem de utilização), levando a uma boa condução.

3.1.1.2.2 Agrupamento/distinção por formato

O agrupamento/distinção por formato refere-se às características gráficas da informação apresentada na tela (formato, cor, etc.) informando se determinado item pertence ou não à determinada classe. Similaridades ou diferenças de relacionamentos são mais fáceis de aprender e lembrar, levando também a uma boa condução.

3.1.1.3 *Feedback* imediato

Refere-se às respostas, às ações dos usuários. Essas respostas podem ser geradas a partir de um pressionar de tecla, até uma lista de comandos. Para isso o usuário deverá ser informado de forma rápida, do passo apropriado a cada tipo de transação. Transações mais longas também devem ser informadas ao usuário

através de evolução do processo.

Qualidade e rapidez são importantes para a satisfação e confiança do usuário, dando a ele uma melhor compreensão do sistema. A ausência de *feedback* pode ser incômoda e dar impressão de falha, podendo desencadear ações prejudiciais ao sistema.

3.1.1.4 Legibilidade

A legiblilidade refere-se às características léxicas, das informações apresentadas na tela, que possam dificultar ou facilitar a leitura (brilho nos caracteres, contraste entre letra e fundo, tamanho da fonte, espaçamento entre caracteres, espaçamento de parágrafos e linhas, e comprimento de linha, etc.). Por definição, legibilidade não engloba mensagens de erro ou de *feedback*. Uma boa legibilidade facilita a leitura da informação, melhorando a performance da tarefa.

3.1.2 Carga de trabalho

A carga de trabalho refere-se a elementos da interface que têm importante papel na redução da carga cognitiva e perceptiva dos usuários e no aumento da eficiência do diálogo. Divide-se em brevidade e densidade informacional.

3.1.2.1 Brevidade

Brevidade corresponde à carga de trabalho perceptiva e cognitiva de entradas e saídas individuais, ou ainda do conjunto de entradas, (por exemplo, conjunto de ações necessárias para atingir determinada meta). Objetiva-se limitar a carga de trabalho de leitura e a entrada de dados e passos para atingir uma determinada meta. Está subdivida em concisão e ações mínimas.

3.1.2.1.1 Concisão

Concisão refere-se à carga perceptiva e cognitiva de entradas e saídas

individuais. A capacidade de memória de curto termo é limitada. Consequentemente, entradas curtas têm menor probabilidade de ocorrência de erros. Além disso, itens mais sucintos serão lidos mais rapidamente.

3.1.2.1.2 Ações mínimas

O critério das ações mínimas diz respeito ao número de ações necessárias par atingir um objetivo ou uma tarefa. Sugere-se limitar tanto quanto possível o número de passos pelos quais o usuário deva passar.

Quanto maior for o número e mais complexas forem as ações necessárias para atingir um objetivo, maior será a carga de trabalho, consequentemente, aumentará o risco de ocorrência de erros.

3.1.2.2 Densidade informacional

A densidade informacional refere-se à carga de trabalho do ponto de vista perceptivo e cognitivo, em relação ao conjunto total de itens de informações apresentadas, e não a cada elemento ou item individualmente.

Em muitas tarefas, a performance do usuário é prejudicada quando a densidade de informações é muito alta ou muito baixa, ocorrendo maior probabilidade de erros. Neste item, devem ser tomados cuidados como a remoção de itens desnecessários à determinada tarefa. Além disso, os usuários não devem ser obrigados a memorizar longas listas de dados e procedimentos complicados.

3.1.3 Controle explícito

O controle explícito refere-se tanto ao processamento explícito pelo sistema a partir de ações do usuário, quanto ao controle que o usuário tem sobre o processamento de suas ações pelo sistema. Divide-se em ações explícitas e controle do usuário.

3.1.3.1 Ações explícitas do usuário

O critério das ações explícitas do usuário refere-se ao relacionamento entre o processamento do computador e as ações dos usuários. Esse relacionamento deve ser explícito, como por exemplo: o sistema deve executar somente através de ações solicitadas pelo usuário, e somente quando ele as solicitar.

Quando o processamento é executado após ações explícitas do usuário, ele aprende e entende melhor o funcionamento da aplicação, e menos erros são observados.

3.1.3.2 Controle do usuário

O controle do usuário refere-se ao fato de que os usuários deveriam estar sempre com o controle do processamento do sistema (por exemplo, interromper, cancelar, parar, continuar, etc.). Cada ação possível deve ser antecipada e opções apropriadas devem ser oferecidas ao usuário. O controle das ações favorece o aprendizado e diminui a incidência de erros, tornando o sistema mais previsível.

3.1.4 Adaptabilidade

Adaptabilidade é o critério que diz respeito à capacidade do sistema em reagir de acordo com o contexto e com as necessidades e preferências do usuário. É subdividido em flexibilidade e experiência do usuário.

3.1.4.1 Flexibilidade

Refere-se à possibilidade de o usuário customizar a interface de acordo com estratégias do seu trabalho, com seus hábitos e requerimentos da sua tarefa. É a capacidade da interface em se adaptar às necessidades particulares do usuário. Quanto mais formas existirem de efetuar determinada tarefa, maiores são as chances do usuário escolher e dominar uma delas durante a aprendizagem.

3.1.4.2 Experiência do usuário

Os mecanismos disponíveis no sistema devem se adequar ao nível de experiência do usuário. Usuários experientes e novatos têm diferentes necessidades de informação. Usuários novatos podem ser conduzidos à execução passo-a-passo, já para os experientes, podem-se oferecer atalhos para que eles alcancem metas já conhecidas. A interface deve estar preparada para lidar com diferentes níveis de experiência e levar em conta que usuários novatos podem tornar-se experientes à medida que utilizarem intensamente a interface. O processo inverso pode ocorrer com usuários experientes que passam longos períodos sem interagir com o sistema.

3.1.5 Gestão de erros

A gestão de erros refere-se aos mecanismos do sistema em prevenir a ocorrência de erros e, caso eles ocorram, possibilitar a sua correção. São considerados erros as entradas inválidas de dados, a sintaxe de comandos errados, o formato de dados errados. Reduzindo a quantidade de incidência de erros, ocorre uma melhor performance por parte do usuário.

Este critério divide-se em proteção contra erros, qualidade das mensagens de erro e correção de erros.

3.1.5.1 Proteção contra erros

A proteção de erros refere-se aos mecanismos empregados para detectar e prevenir erros de entradas de dados ou comandos. Prevenindo erros durante a digitação muito mais do que no momento da validação evita perturbações durante a execução da tarefa.

3.1.5.2 Qualidade das mensagens de erros

Diz respeito ao conteúdo das mensagens de erro, sua relevância, severidade, legibilidade, exatidão da informação dada ao usuário com relação ao erro cometido e sobre as ações que ele poderá tomar para corrigi-lo.

A qualidade das mensagens de erro favorece o aprendizado do sistema, à

medida que ele informa a razão pelo qual o erro ocorreu e a forma de corrigi-lo.

3.1.5.3 Correção de erros

Refere-se à possibilidade do usuário corrigir erros cometidos. Quando os erros podem ser facilmente reparados tornam-se menos perturbadores para os usuários.

3.1.6 Consistência

O critério consistência refere-se às formas escolhidas para conceber interfaces (códigos, nomes, formatos, procedimentos, etc.), quando elas são conservadas idênticas em contextos idênticos, e diferentes para contextos diferentes.

Procedimentos, rótulos e comandos são melhor localizados, reconhecidos e utilizados, quando seu formato, localização ou sintaxe são estáveis de uma tela para outra, ou de uma sessão para outra. Dessa forma o sistema torna-se mais previsível, facilita o aprendizado, diminui a ocorrência de erros e dá ao usuário sensação de estabilidade diante das interfaces.

3.1.7 Significado dos códigos

O significado dos códigos e denominações diz respeito à adequação entre termos, ou a unidade de informação e a codificação usada para representá-la. Códigos e nomes são significantes quando há uma forte relação semântica entre os códigos e os itens ou ações representadas.

Se a codificação é significativa para o usuário, então ele recorda e reconhece melhor essas denominações e evita a execução de operações inadequadas, evitando erros.

3.1.8 Compatibilidade

Compatibilidade refere-se à adequação entre características de tarefas do

usuário (memória, percepção, experiência, habilidade, idade, expectativas, etc.) e a organização das saídas e das entradas, dos diálogos de uma dada aplicação para outra. Diz respeito, também, ao grau de similaridade entre diferentes ambientes e aplicativos.

A transferência de informações entre contextos diferentes é mais rápida e eficaz, quando o volume de informações a ser recodificada é menor.

A eficiência é ampliada quando: os procedimentos que acompanham a tarefa são compatíveis com características psicológicas dos usuários; procedimentos e tarefas são organizados com respeito às expectativas e práticas dos usuários; transações, interpretações ou referências para documentação são minimizadas.

Complementando o estudo realizado por Medeiros (1999), encontraram-se recomendações de Tagnazzini, que são inicialmente formuladas para uso em aplicações na web, mas que podem ser perfeitamente aproveitadas para aplicações computacionais tradicionais.

3.2 PRINCÍPIOS DE TOGNAZZINI (2001)

Tognazzini (2001) descreve dezesseis princípios que considera fundamentais para serem aplicados na construção de interfaces com aplicações na *web*. Porém estes princípios também podem ser aplicados em sistemas comuns. Algumas recomendações são específicas para uso de sistemas baseados em *browsers*. Ele justifica: “não é porque uma aplicação ou um serviço apareça na web que os princípios devam ser mudados”.

Para o autor, interfaces eficazes são evidentemente claras e fornecem ao usuário um senso de controle. Usuários facilmente vêem a extensão de suas opções, alcançam seus objetivos e fazem seu trabalho. É importante que o sistema esteja atento ao salvamento do trabalho do usuário e lhe dê opções completas de desfazer alguma ação a qualquer momento. Aplicações e serviços eficazes desempenham um máximo de trabalho, enquanto requerem o mínimo de esforço e

informação do usuário.

Os princípios classificados por Tognazzini são: antecipação, autonomia, cor e brilho, consistência, padrões, capacidade do usuário, interfaces exploráveis, leis de *fitt*, objetos de interface, redução de latência, aprendizagem, uso de metáforas, proteção do trabalho do usuário, legibilidade, histórico do caminho e navegação visível.

Antecipação: aplicações de computador devem se antecipar às necessidades dos usuários, não esperando que o usuário procure por determinadas informações ou busque ferramentas necessárias, e sim lhe oferecendo todas as informações necessárias e ferramentas que ele precisa a cada passo do processo.

Autonomia: a interface, o ambiente e o computador "pertencem" ao usuário, mas essa autonomia não significa abandonar determinadas regras. O usuário deve ter liberdade para tomada de decisão, tendo a sensação de estar dominando a tarefa num espaço explorável mas não perigoso. Deve ainda utilizar-se de mecanismos de *status*, para manter o usuário informado e atento. Onde nenhuma autonomia pode existir sem a ausência de controle, e não se pode oferecer controle sem informação suficiente. O usuário deve ser mantido informado do status do sistema através de mensagens e ícones de fácil interpretação.

Cor e brilho: devem ser usadas cores para diferentes tipos de informações, levando-se em conta que algumas pessoas não identificam determinadas cores, e/ou algumas cores não são suportadas em determinadas configurações. Sugere-se utilização de escalas de cinza e formatação de texto deferenciada para diferentes cores ou rótulos.

Consistência: uma aplicação que obedece níveis de consistência melhora a qualidade do produto e respeita as necessidades do usuário. Porém, algumas semelhanças de alguns ícones podem levar a atos diferentes dos previstos, por isso é importante averiguar as expectativas dos usuários através de testes de campo, utilizando simbologias que lhes são conhecidas. Não devem ser usados ícones parecidos para ações diferentes.

Padrões: os campos com dados padrões devem ser oferecidos ao usuário, podendo os campos serem substituidos quando necessário de forma rápida e fácil.

Capacidade do usuário (eficiência do usuário): é necessário priorizar a produtividade do usuário facilitando a entrada de dados de forma a requerer menos processamento cognitivo. O usuário deve se manter ocupado e produtivo em um sistema eficiente que alcance as metas propostas.

Interfaces exploráveis: oferecer ao usuário interfaces seguras e com marcações de estados nos quais elas se encontram. Não se deve oferecer um único caminho para atingir uma meta. Em uma tarefa de uso único, deve-se oferecer interface dirigida. Deve-se oferecer ao usuário elementos visuais estáveis, permitindo que navegue de forma rápida com marcos seguros, proporcionando a sensação de familiaridade com o ambiente. É importante o fornecimento de ações reversíveis para desfazer ações ou retornar ao estado anterior. Se não for possível oferecer condições de desfazer algo, devem ser implemetadas questões de confirmação para evitar o erro. Deve haver sempre um recurso de saída, com o qual os usuários possam a qualquer momento abandonar o sistema sem se sentirem presos a ele. O usuário deve, isso sim, sentir-se atraído a ficar no ambiente.

Leis de *Fitt*: o tempo para atingir um objetivo está relacionado com a distância e o tamanho deste objetivo. Ícones grandes devem ser usados para funções importantes.

Objetos de interface: é importante o uso de objetos cotidianos para representar objetos de interface da aplicação, com os quais o usuário tenha familiaridade.

Redução de latência: sempre que possível, a latência deve ser reduzida, usando-se de multiprocessamento para que o usuário consiga desempenhar outra tarefa enquanto aguarda algum resultado do sistema. É útil a exibição de ampulheta animada, indicando que o sistema ainda está ativo. A duração de processamentos longos deve ser indicada por mensagens. Deve-se comunicar a duração atual através de um indicador de progresso animado. Alarmes com indicações visuais grandes devem informar ao usuário que ele pode retomar o comando do sistema. É importante o desenvolvimento de aplicações rápidas, eliminando informações e processamentos desnecessários.

Aprendizagem: as aplicações, por mais simples que pareçam, sempre

oferecem graus de aprendizado de uso.

Uso de metáforas: o uso de metáforas permite ao usuário compreender o modelo conceitual mais rapidamente.

Proteção do trabalho do usuário: o usuário deve ter a segurança de nunca perder o trabalho por sua falha ou por queda de energia.

Legibilidade: o que deve ser lido, deve ter destaque. A formatação de fontes deve estar em tamanho legível, levando em conta características de pessoas com necessidades especiais, como as mais velhas, dautônicas ou com alguma deficiência visual.

Histórico do caminho: o usuário deve manter-se informado de onde ele está, por onde ele passou, e aonde está indo, possibilitando que volte facilmente a um determinado ponto já visitado, ou mesmo ao ponto inicial.

Navegação visível: auxiliar o usuário a construir mapas mentais para não ter a sensação de estar perdido no *site*, tornando a navegação clara e natural, é muito importante.

3.3 HEURÍSTICAS DE NIELSEN (2002)

A seguir, são listadas e descritas brevente as dez regras heurísticas derivadas por Nielsen (1994), com colaboração de Rolf Molich (MOLICH e NIELSEN, 1990; NIELSEN e MOLICH, 1990 apud NIELSEN 2002). As heurísticas auxiliam na avaliação e desenvolvimento de interfaces interativas.

Visibilidade do status do sistema: o sistema deve sempre manter o usuário informado sobre onde está indo, dando o *feedback* dentro de tempo razoável.

Aproximação entre sistema e o mundo real: o sistema deve falar o idioma dos usuários, com palavras, frases e conceitos familiares ao usuário, ao invés de termos técnicos. Deve seguir convenções do mundo real, fazendo a informação aparecer na ordem natural e lógica.

Controle do usuário e liberdade: os usuários escolhem freqüentemente, por

engano, funções de sistema e precisarão de uma “saída de emergência” para abandonar a operação sem ter que passar por um diálogo extenso. Devem-se apoiar ações de desfazer (*undo*) e refazer (*redo*).

Consistência e padrões: usuários não devem precisar saber que palavras diferentes, situações ou ações significam a mesma coisa. Por isso, convenções de plataforma devem ser seguidas.

Prevenção de erro: sempre melhor do que boas mensagens de erro é o cuidadoso projeto de interfaces que previnam um problema antes que ele aconteça.

Reconhecimento ao invés de recuperação: deve-se primar por objetos, ações, e opções visíveis. O usuário não deve ter que se lembrar de informação de uma parte do diálogo para outro. Instruções para uso do sistema deveriam ser visíveis ou facilmente recuperáveis sempre que apropriado.

Flexibilidade e eficiência de uso: aceleradores não vistos por usuários principiantes podem acelerar freqüentemente a tarefa de usuários especialistas, de forma que o sistema possa suprir necessidades tanto para novatos como para usuários experientes, além de permitir que os usuários construam ações freqüentes.

Design claro (estético) e minimalista: diálogos não devem conter informações irrelevantes ou raramente utilizadas. Toda unidade extra de informação em um diálogo se confunde com as unidades pertinentes de informação e diminui a visibilidade das informações realmente pertinentes.

Reconhecimento de ajuda a usuários, diagnósticos e recuperação de erros: mensagens de erro devem ser expressas em linguagem clara (não em códigos), indicando exatamente o problema e sugerindo uma solução, além de ser sensíveis ao contexto.

Ajuda e documentação : embora seja melhor que o sistema possa ser usado sem documentação, pode ser necessário prover ajuda e documentação. Tais informações devem ser fáceis de encontrar e focadas na tarefa do usuário. É importante listar passos concretos e não muito extensos a serem desempenhados.

Nielsen (2002) recomenda também a leitura das lista de Tognazzini de princípios básicos para design de interface, que, embora bastante extensa, serve

como um poderoso *checklist*.

3.4 CONCLUSÃO COM RELAÇÃO AOS CRITÉRIOS E RECOMENDAÇÕES ERGONÔMICAS

As recomendações ergonômicas de Scapin e Bastien (1997) são organizadas de forma hierárquica e concisa, o que facilita sua aplicação em avaliações ergonômicas, e estas serão utilizadas nessa dissertação. Os princípios de Tognazzini (2001) apesar de terem sido organizados para construção de sistemas baseados em *browsers*, mostram-se bastante pertinentes ao uso em quaisquer sistemas com interfaces interativas ou em suas avaliações. Já as heurísticas de Nielsen (2002) em conjunto com as recomendações Scapin e Bastien (1997), demonstraram-se bastante úteis na avaliação ergonômica efetuada nessa dissertação, e por este motivo encontram-se aqui relatadas.

4 TIPOS DE AVALIAÇÕES ERGONÔMICAS E MÉTODOS DE INSPEÇÃO E SUA CLASSIFICAÇÃO

Os objetivos da avaliação ergonômica de *software*, entre outros são a otimização da qualidade de interação homem-máquina, da satisfação do usuario, a correção de erros e do aumento de produtividade.

Muitos programas computacionais, ao invés de facilitar a vida do usuário, criam verdadeiras ciladas: executam funções sem solicitar sua confirmação, dificultam a entrada de dados, levam a erros no encaminhamento das tarefas. Casos extremos, somados a problemas e distúrbios pessoais, podem provocar uma situação que os especialistas chamam de “loucura do trabalho”. Esses casos certamente são raros, geralmente os problemas relacionam-se à irritação e à insatisfação, mas estas também são situações que devem ser resolvidas (CYBIS, 1998).

Nas interfaces homem-computador, a ergonomia de *software*, através do conhecimento das capacidades, limites e outras características do desempenho humano, preocupa-se em projetar, avaliar e melhorar as aplicações informatizadas, para aumentar a segurança, o conforto e a eficiência do sistema e da qualidade de vida do usuário. A ergonomia de *software* se interessa, ao mesmo tempo, pela adequação à tarefa (utilidade), a facilidade de uso (usabilidade) e a utilizabilidade (usabilidade + utilidade) dos produtos e sistemas informatizados para atingir mais facilmente seus objetivos. A utilidade verifica se o produto ou sistema atende às necessidades funcionais e operacionais, e a utilizabilidade diz respeito à qualidade que a interface possui, para facilitar a utilização e a aprendizagem (SENACH apud SILVA, 2002).

O usuário “vê” o sistema através da interface, caso elas sejam inadequadas, o *software* pode ser abandonado, mesmo tendo ótimas funcionalidades (MOÇO, 1996).

A avaliação de humano-computador interfaces deve ser um processo totalmente integrado nas fases de concepção e implementação de um produto. Nas fases de concepção, a avaliação deve prever a usabilidade do produto, verificar a compreensão dos requisitos do usuário pela equipe de desenvolvimento, através de prototipação rápida e informal de idéias. Nas fases posteriores à concepção, a avaliação auxilia na identificação das dificuldades dos usuários, e auxilia também na análise de custo/benefício, apontando aspectos que devem ser melhorados em versões futuras do sistema. “Testes de usabilidade são desenvolvidos com a finalidade de encontrar erros graves e reduzir os gastos com grandes correções nos projetos em desenvolvimento” (MOÇO, 1996, p. 16).

Nesse aspecto, Moço (1996 p. 10) declara ainda:

> (...) fazer análise ergonômica de tarefas, onde movimentos e ações são visíveis, não é tarefa fácil. Fazer análise de programas de software interativo é ainda mais difícil. Havendo predominância do trabalho cognitivo sobre o trabalho motor, detectar o processo operante de uma pessoa em situação real de trabalho, interagindo com um aplicativo, requer muito preparo, técnica e habilidade por parte do analista.

Problemas de usabilidade podem ser classificados de acordo com o tipo de usuário que afetam, ou com o tipo de tarefa em que se manifestam. O problema pode ser geral, quando afeta qualquer tipo de usuário; inicial, quando atinge apenas usuários inexperientes; avançado, quando compromete a realização de tarefas executadas por usuários experientes; e especial, quando atinge um grupo específico de usuários (por exemplo, portadores de alguma deficiência). Quanto à tarefa, o problema pode ser classificado como principal, quando compromete tarefas importantes e freqüentes, e secundário quando atinge tarefas opcionais ou de menor importância.

4.1 ESCOLHA DO MÉTODO/TÉCNICA

Muitas teorias são requeridas para descrever os múltiplos aspectos dos sistemas interativos. Algumas teorias são exploratórias, elas ajudam na observação do conhecimento, na descrição das atividades, na concepção do *design*, e também na comparação de projetos e treinamento. Outras teorias são preditivas, estas habilitam os *designers* a comparar as propostas de *design*, para executar no tempo, e as taxas de erros. Algumas teorias podem focar no perceptivo ou em subtarefas cognitivas (tempo para encontrar um item na tela, ou tempo para planejar a troca de uma fonte em um texto). Outras teorias se concentram em tempos de performances de tarefas motoras. Predições de tarefas motoras são melhor estabelecidas e são acuradas por captura de teclados ou tempos de apontamento. É difícil predizer tarefas cognitivas complexas, porque podem tomar seqüências diferentes, dependendo do entendimento do usuário, ou por ocorrência de erros. É diferente a performance de usuários novatos e *experts*; alguns novatos e usuários iniciantes podem não alcançar os objetivos ou não completar a tarefa (SHNEIDERMAN, 1998).

Para se efetuar uma avaliação, são necessárias técnicas especiais, tempo para esperar a melhor oportunidade de interagir, paciência por parte dos ergonomistas, considerar custos não previstos no orçamento e contar com a boa vontade de usuários que queiram participar e contribuir com os testes.

Para escolha do método adequado de avaliação de *software*, Schneiderman (1998, p. 124) aconselha o estudo de alguns fatores, que são:

a) estágio em que se encontra o projeto: indica em que fase de desenvolvimento encontra-se o sistema a ser analisado, fases iniciais, em andamento ou em fase de conclusão;

b) grau de inovação do projeto: indica se o domínio do assunto tem natureza bem definida ou se é de caráter exploratório;

c) número de usuários potenciais;

d) criticidade da interface: indica se o sistema refere-se a algo de uso crítico, um exemplo seria a criticidade de uma interface de um controle de reator nuclear ou de tráfego aéreo, diferentemente de um quiosque de informações turísticas;

e) custo do produto e dos recursos destinados a testes: há métodos de inspeção cujos custos e precisão são bastante variados. A criticidade do produto e o grau de precisão dos resultados esperados dos testes serão significativos para a busca do equilíbrio entre os custos de desenvolvimento do produto e os custos de inspeção da usabilidade;

f) disponibilidade de tempo e de especialistas em usabilidade;

g) experiência da equipe de projeto e avaliação.

Moço (1996) acrescenta também, que toda técnica tem suas limitações, cabendo ao ergonomista ter o bom senso de escolher e adaptar a técnica ao trabalho que realizar, levando em consideração outros pontos, como:

a) local onde ocorrerá a avaliação;

b) as características do usuário que participará das avaliações;

c) avaliação dos benefícios que a técnica escolhida pode trazer para o projeto.

Método de inspeção, ou de avaliação, é um nome genérico para um conjunto de métodos baseados em inspetores, examinando aspectos relacionados com a interface do usuário. Inspetores podem ser especialistas de usabilidade (ergônomos), mas também podem ser programadores com habilidades especiais, usuários finais ou conhecedores da tarefa, ou outro tipo de profissional. Inspeções também podem ser executadas mesmo que o sistema ainda não exista, ou seja, em fases iniciais do ciclo de vida do sistema. O atributo mais importante em um método de avaliação de usabilidade é a qualidade dos dados que ele gera. O sucesso da aplicação de um método de avaliação está relacionado com a qualidade dos dados obtidos; estes precisam ser úteis para prover ao analista informações corretas sobre os problemas encontrados (NIELSEN, 2002).

Diversos tipos de avaliações e metodologias estão disponíveis e são classificadas de várias maneiras, por diversos autores; todas trazem vantagens e desvantagens. Alguns estudos já se dedicaram a comparar esses métodos, que quando pertinentes serão aqui relatados; em outras situações serão somente descritas suas características, para auxiliar na classificação do método utilizado

neste estudo. Autores como Moço (1996), Shneiderman (1998), Jorge (2002), Hole (2002), Cybis (1997) e Nielsen (apud MEDEIROS, 1999) classificam os tipos de avaliação de interfaces com nomenclaturas pouco diferentes, e na sua essência todos tendem à classificação conforme o quadro a seguir. Basicamente os tipos de avalição e as técnicas utilizadas podem ser classificados utilizando-se, ou não, da participação do usuário. As técnicas que envolvem a participação do usuário podem ser classificadas em duas categorias, que são: as prospectivas, que envolvem a opinião do usuário, e as objetivas/empíricas/interpretativas, ainda chamadas de etnográficas, que contam com a participação direta dos usuários em situações de uso do produto. As que não necessitam da participação dos usuários são as do tipos preditiva/analítica, ainda definidas também como revisões especializadas.

Técnicas preditivas/analíticas dispensam a participação direta de usuários nas avaliações e inspeções. Baseiam-se nos conhecimentos ergonômicos e experiência dos avaliadores. O sucesso da aplicação do método depende diretamente da carga de conhecimento do ergonomista, agregada à ferramenta de inspeção utilizada. Geralmente, os avaliadores que adotam esses métodos são especialistas em usabilidade ou projetistas de sistemas. Os avaliadores normalmente se baseiam em regras, recomendações, princípios e/ou conceitos previamente estabelecidos e definidos por diversas áreas de pesquisa. Essas regras contribuiem nas inspeções auxiliando na identificação de problemas de usabilidade que provavelmente afetam (ou afetarão) as interações dos usuários reais com o sistema. Além disso o avaliador precisa estar atento também ao contexto onde se encontra a aplicação a ser avaliada.

Para Jorge (2002), no tipo de avaliação interpretativa, o objeto da avaliação já existe. Nesse caso, avaliam-se os resultados, para determinar o que está correto, o que está errado, e o que precisa ser melhorado. Na avaliação preditiva, quando o objeto ainda não existe, avaliam-se os requisitos e as especificações, fazendo comparações entre especificações e modelo de requisitos.

Técnicas empíricas contam com a participação direta de usuários, sendo a principal técnica dessa categoria os ensaios de interação (CYBIS, 1997).

Com relação à classificação dos tipos de avaliação efetuada, Shneiderman

(1998) ainda faz referência quanto à forma de condução, o ambiente e os envolvidos na avaliação (especialistas, usuários ou ambos).

Métodos podem ser usados em conjunto para aumentar a detectação de diferentes tipos de erros.

QUADRO 1: SÍNTESE CLASSIFICATÓRIA DE TIPOS DE AVALIAÇÃO E TÉCNICAS UTILIZADAS

TIPO DE AVALIAÇÃO	***OS ENVOLVIDOS***	***TÉCNICA/MÉTODO UTILIZADOS***
Preditiva/ analítica (revisões especializadas)	Sem a participação do usuário (Baseadas nos conhecimentos ergonômicos e experiência dos avaliadores)	Avaliações heurísticas Inspeções via *checklists* Inspeções cognitivas Inspeções formais Inspeções de consistência Inspeções baseadas em padrões Avaliação de documentação
Prospectiva	Opinião do usuário	Entrevistas e questionários
Objetivas/empíricas/interpretativa (etnografia)	Com a participação do usuário	Observações da tarefa Ensaios de interação e cenários Testes de uso ativo Captura por sistemas espiões

A seguir serão descritas as técnicas/métodos de avaliação de *software*, conforme classificação do quadro 1.

4.2 AVALIAÇÕES PREDITIVAS/ANALÍTICAS

As avaliações preditivas, também denomidadas como analíticas ou revisões especializadas, envolvem as técnicas de avaliação heurísticas, inspeções via *checklist* ou *guidelines* e inspeções cognitivas .

4.2.1 Avaliações heurísticas

Avaliação heurística é um método informal de engenharia de usabilidade, que é rápido, barato e de fácil aplicação em uma avaliação de interface. A avaliação heurística é um método de inspeção bastante popular. É feito com a inspeção sistemática da interface para verificar a usabilidade. O objetivo desta avaliação é encontrar problemas de usabilidade no *design*, tanto que ela é parte de um processo interativo de *design*. Envolve um conjunto pequeno de avaliadores examinando a interface e julgando-a através de princípios reconhecidos de usabilidade, as heurísticas. (NIELSEN, 2002).

Embora considerado um método simples e relativamente rápido, o método requer conhecimento do avaliador, para aplicação das heurísticas. Porém, Nielsen (2002) relata que resultados experimentais mostram que a heurística é difícil para um único indivíduo, porque uma pessoa dificilmente encontrará todos os problemas da usabilidade em uma interface. Felizmente, a experiência de muitos projetos diferentes mostrou que pessoas diferentes encontram problemas diferentes de usabilidade, indicando que o número ideal de avaliadores está entre 3 e 5 pessoas.

Devido ao alto grau de subjetividade deste tipo de avaliação, cujos resultados dependem da experiência e dos conhecimentos acumulados pelo ergonomista e das estratégias utilizadas pelo avaliador, sugere-se, como forma de uniformizar as análises e garantir uma média de desempenho individual, a aplicação do conjunto de critérios ergonômicos como ferramenta de análise (RIBEIRO, 2002).

A análise heurística é um método significativo para achar tanto problemas principais (maiores), como problemas secundários (menores) em uma interface. Problemas principais são ligeiramente mais fáceis de serem encontrados que problemas secundários. Um problema principal foi encontrado por um único avaliador em média de 42% em 6 casos em que o autor aplicou o método. Já a probabilidade de encontrar um erro secundário foi de 32% (NIELSEN, 2002). Embora problemas principais sejam mais fáceis de encontrar, não significa que avaliadores se concentrem exclusivamente neles. Os problemas principais são por definição mais importantes de achar e isolar, mas problemas secundários também

são pertinentes. Nielsen (2002) relata que muitos dos problemas secundários parecem ser mais fáceis de serem encontrados através de modelos heurísticos.

A avaliação heurística apoiada através de critérios ergonômicos apresenta bons resultados em termos de ferramentas de análise.

4.2.2 Inspeções via *checklists* (g*uidelines*)

Geralmente a avaliação via *checklist*, também denominada como lista de verificação, ou *guidelines*, é associada a outros métodos de inspeção, como avaliações heurísticas, ou inspeção de consistência. Os guias, ou as listas, são usados pelos avaliadores como roteiro e/ou conjunto de requisitos, critérios ou princípios básicos, que são considerados desejáveis e/ou necessários na interface, sendo verificados no transcorrer da avaliação. Esse método facilita a identificação de problemas, reduz o nível de subjetividade, e reduz os custos. Uma vez que já possuem embutidos conhecimentos ergonômicos, podem ser aplicados por outras pessoas, que não sejam especialistas em usabilidade e ergonomia.

Existe diferenciação entre guias de recomendações e guias de estilos. Estilos são publicações com descrições detalhadas de elementos interativos específicos de um sistema (menus, janelas, caixas de entradas de dados, etc.). Normalmente são elaborados em uma organização com o objetivo de estabelecer padrões, convenções e modelos, para melhorar a consistência dos sistemas. Os guias de recomendações são documentos públicos que relatam recomendações geradas e validadas a partir de observações empíricas e experiências práticas do autor.

4.2.3 Inspeções cognitivas (*cognitive walkthrough*)

Usa um procedimento explicitamente detalhado com o objetivo de simular o processo que o usuário desempenha para resolver problemas em cada etapa do diálogo, verificando se os objetivos do usuário simulado e se a carga cognitiva puderam ser supostos para conduzir a ação correta para completar a tarefa (NIELSEN, 2002).

Trata-se de um modo formalizado de imaginar os pensamentos e ações dos usuários leigos ao utilizar as interfaces pela primeira vez, podendo também introduzir teorias psicológicas dentro da técnica informal e subjetiva de exploração cognitiva (RIBEIRO, 2002).

4.2.4 Inspeções formais

É uma variação da avaliação de usabilidade da metodologia tradicional de inspeção de *software*. Combina inspeções individuais e em grupos com procedimentos e regras estritamente definidos com elementos de avaliação heurística e *cognitive walkthroughs*. Esse método é geralmente adotado nas fases preliminares de desenvolvimento de sistemas, para detectar defeitos ou problemas de usabilidade. Também são conhecidas como inspeções de código, para detectar erros no *software*, que no jargão de informática são chamadas de *bugs* (DIAS, 2002).

4.2.5 Inspeções de consistência

Designers que fazem parte de outros sistemas inspecionam a interface em relação a um modelo próprio para comparar se as aplicações estão dentro de um mesmo padrão, garantindo assim a consistência com outras aplicações. Este método é mais usado nas fases preliminares de desenvolvimento. Hom (1996 apud DIAS, 2002) considera que o momento ideal para aplicação desse método é quando a especificação de cada sistema individual já está praticamente pronta, antes do início efetivo de seu desenvolvimento como produto.

4.2.6 Inspeções baseadas em padrões

Este tipo de inspeção verifica a conformidade do sistema ou produto com relação a algum padrão, através de confrontação, seja ele uma norma estabelecida por organismos internacionais como ISO e IEEE (Institute of Electrical and Eletronics Engineers, IEEE), normas técnicas nacionais como ABNT (Associação Brasileira de Normas Técnicas) ou institutos norte-americanos como ANSI (American National Standard Institute) e NIST (National Institute of Standard and Technology) ou outros

parâmetros como os do fornecedor do sistema operacional (DIAS, 2002).

4.2.7 Avaliação de documentação

Apesar de não ser considerado como uma técnica em si, o estudo e avaliação da documentação existente pode ser uma boa fonte de informações a respeito do sistema a ser avaliado. Pode mostrar a estrutura e os objetivos a que o *software* se destina e traz uma visão inicial e geral ao avaliador antes da aplicação de outro método.

Para Dix (1993), a leitura desses materiais é importante e auxilia o analista a ter uma visão mais ampla do sistema e da organização antes de efetivar a observação propriamente dita. Declara contudo, que as documentações podem ser fontes incompletas, pois muitas vezes descrevem as “supostas” formas de desempenho das tarefas e não como realmente as pessoas trabalham. Para isso a observação *in loco* se faz necessária, para confrontar com essas informações e obter dados reais.

4.3 AVALIAÇÕES PROSPECTIVAS

As avaliações prospectivas contam com a participação do usuário através de informações fornecidas por meio de questionários ou opiniões pessoais a respeito do sistema, são também denominadas como pesquisas de opinião (do termo em inglês, *survey evaluation*). Estas informações são importantes para a avaliação de sistemas e podem ser coletadas através de entrevistas e questionários.

As entrevistas podem ser divididas em entrevistas estruturadas e entrevistas flexíveis (ou abertas). Nas entrevistas estruturadas existem pré-determinadas áreas subjetivas e conjunto de perguntas. São fáceis de analisar, mas algum assunto pode ser esquecido pelo analista. Entrevistas flexíveis não possuem uma seqüência fixa de perguntas, apenas um guia que permite elaboração de novas questões durante o seu transcorrer. É possível obter respostas discursivas dos usuários sobre determinados assuntos. A análise nesse formato consome mais tempo. As

entrevistas são consideradas técnicas informais, são importantes para obter informações dos usuários a respeito do sistema, suas ansiedades e satisfação. Mas, por outro lado, tornam difícil a aferição da confiabilidade e validade de seus resultados.

As entrevistas diferenciam-se de questionários, pois são feitas através de contato direto entre entrevistado e entrevistador. O questionário puro pode ser aplicado sem a participação direta do entrevistador. As informações coletadas por esse método podem ser posteriormente tabuladas e analisadas estatisticamente.

Para Gonçalves (2002), os questionários podem funcionar com perguntas fechadas (com respostas “sim”, “não”, “não sabe/não responde”; escala e ordenação). Uma das vantagens para uso de questionários fechados é o processamento estatístico, com médias, desvio padrão, significância. Hole (2002) declara que esta análise é considerada relativamente simples. Como desvantagens, têm-se o custo e a probabilidade de respostas muito variadas. Questionários são úteis para obter informações quando existir um grande número de usuários, ou estiverem geograficamente dispersos, segmentados por perfis, ou por uma amostragem. Podem-se obter indícios de problemas por um conjunto de usuários, que depois poderão ser reinvestigados através de outros métodos.

4. 4 AVALIAÇÕES OBJETIVAS/EMPÍRICAS/INTERPRETATIVAS

As técnicas utilizadas nas avaliações objetivas, também denominadas empíricas ou interpretativas, compreendem as observações da tarefa ou análise da tarefa simulada, os ensaios de interação com uso de cenários, teste de uso ativo e captura por sistemas espiões.

4.4.1 Observações da tarefa ou análise da tarefa

A análise da tarefa envolve componentes básicos, como o planejamento, a coleta e análise de dados oriundos de observações. O processo de coleta de dados

é feito através da observação da tarefa.

Observar uma tarefa qualquer não é atividade fácil, observar uma tarefa envolvendo uma aplicação HCI é particularmente mais difícil (MOÇO,1996). Para diminuir a diferença entre o mundo real e o mundo percebido é preciso associar disciplinas como a psicologia, filosofia e ciências cognitivas.

Em HCI, em que são desempenhadas tarefas usando uma estação de trabalho, com vídeo, teclado, mouse e outros dispositivos, o conhecimento gerado pelo operador é muitas vezes rápido demais, sendo não somente difícil como usualmente impossível observar o usuário digitando e observando o que surge na tela, simultaneamente. Para aumentar a qualidade dessas observações, é preciso a utilização de filmagens para manter vivas as observações da tarefa para serem revisadas e analisadas posteriormente.

O relato de uma observação, seja ela verbal, escrita ou uma descrição gráfica de uma observação visual, é sempre uma pobre sombra da rica percepção do observador. Para Popper (1988 apud DIAPER, 1989), não há descrição de dados livres da linguagem, e essa interferência é esperada.

Como sugestões alternativas para a observação da tarefa está o uso de protocolos verbais concorrentes, através dos quais o usuário relata verbalmente o que ele está fazendo durante o desempenho da tarefa. Porém, esse paliativo acaba interferindo no desempenho da tarefa.

As observações podem ser ativas, quando o observador segue a observação fazendo perguntas ao usuário, ou passivas, quando o analista apenas observa o usuário. Posteriormente, pode-se usar a técnica de *post-task walkthrough* na qual o observador questiona o usuário sobre as ações.

Outro ponto importante é a definição da tarefa a ser observada. É mais importante identificar o propósito da observação da tarefa e a subsequente análise a identificar o objetivo do desempenho da tarefa em si. Deve-se selecionar o que será observado, o mais importante é decidir o que observar.

A fidelidade das informações adquiridas através de observações também deve ser levada em conta. É sabido que uma técnica de observação, por mais bem

preparada e corretamente aplicada, será somente uma “sombra” do que acontece de fato, e de como esse acontecimento é percebido e relatado pelo observador.

A observação pode ser informal, quando é realizada no campo, ou seja, no local de trabalho do usuário, ou formal, quando é realizada em laboratório através de observações em testes de usabilidade.

Observar potenciais usuários em seu ambiente natural, resulta em uma percepção mais precisa do problema do que somente perguntar aos usuários o que eles fazem. São informações úteis para suporte à automação de uma função pouco ou não automatizada, particularmente quando são disponíveis a observadores treinados sem noções preconcebidas do problema e de sua solução (CARVALHO et al., 2002).

Na observação direta, o observador toma notas do comportamento do utilizador e regista o seu desempenho. Esta é considerada uma técnica instrusiva e subjetiva. Uma das maiores dificuldades é em relação à tomada de decisão, pelo observador, em definir o que é importante ser observado. Quando a observação tem natureza indireta, são efetuados registros em vídeo, que podem conter outros registros simultâneos (uso de *softwares* espiões). A dificuldade encontrada nessa categoria é a complexidade e demora na análise, distanciamento entre os utilizadores e os observadores, ocorrendo perdas de informações, além de requer planejamento prévio (utilização de câmeras, etc.) (JORGE, 2002).

4.4.2 Captura por sistemas espiões

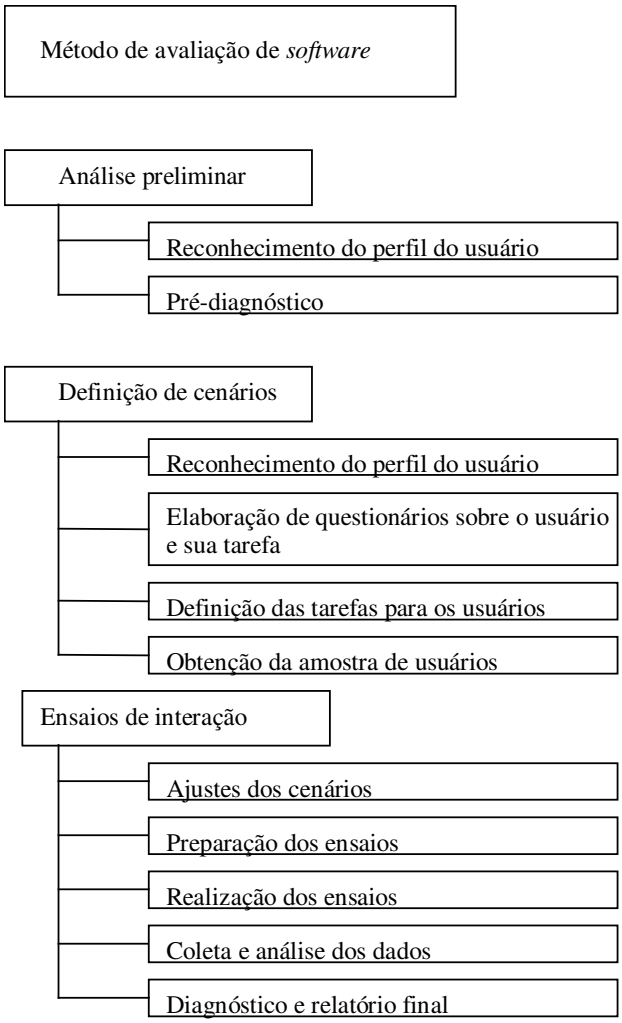
Softwares espiões são sistemas que permanecem ativos no computador durante a execução de tarefas. Esses registram as movimentações de *mouse* e alterações dos estados ocorridos na interface do *software* em tempo real. Essas gravações poderão ser reproduzidas posteriormente, para análise das inferências do usuário. Esse recurso também pode ser associado a um outro método para enriquecer a coleta de dados. O registro por esses *softwares* espiões geralmente gera arquivos de tamanho bastante grande, e isso deve ser levado em consideração no caso da necessidade da transferência desses registros a outras máquinas. Este recurso pode ser utilizado com outros auxílios, como gravação de imagens e voz

simultaneamente, para enriquecer a coleta de dados. Possui plasticidade na análise, devido ao grande volume de informações que gera.

Essa categoria, para Gonçalves (2002), tem vantagem de recolher a ação verdadeira, e o autor classifica como sendo do tipo não-intrusiva, mas levanta problemas éticos caso os usuários não sejam avisados dessa conduta. Outra desvantagem pontuada pelo autor é o grande volume de informações que tende a ter análise difícil e custo elevado.

4.4.3 Ensaios de interação/testes de usabilidade e cenários

Segundo Lewis e Reiman apud Moço (1996, p. 39), “teste de usabilidade implica em pessoas reais tentando fazer tarefas reais com um sistema real e com um observador real vendo o que acontece”. Os testes de usabilidade são um conjunto de atividades que são executadas após detalhada preparação. Esse processo é bem definido por Cybis (1997), e será descrito nos tópicos abaixo. Seu resumo pode ser visualizado na figura a seguir.



FONTE: (CYBIS, 1997, p. 82)

FIGURA 6: ETAPAS DE AVALIAÇÃO ERGONÔMICA DE *SOFTWARE*

4.4.3.1 Análise preliminar

O contexto de desenvolvimento é estudado,sendo feito o contato com projetistas e o reconhecimento prévio do *software*.

4.4.3.2 Reconhecimento do *software*

O reconhecimento do software tem como objetivo compreender o ciclo de desenvolvimento pelo qual o *software* passou ou está passando, para a partir desse conhecimento fazer um pré-diagnóstico do produto.

Esse reconhecimento pode ser feito através de entrevistas com várias

pessoas envolvidas na construção do *software*, sejam elas projetistas, desenvolvedores e/ou gerentes. Deve-se tentar com essa aproximação descobrir os objetivos que eles pretendem com a avaliação ergonômica.

Podem ser solicitadas da equipe de projeto informações sobre o tempo gasto no projeto, se houve interrupções e por quê; quantas pessoas foram envolvidas no desenvolvimento; se o projeto sofreu alterações significantes desde sua concepção; a quem se destina o *software*; quais são as características dos usuários principais; que tarefas o aplicativo pretende auxiliar; qual o impacto na tarefa do usuário; e se o *software* vai alterar a rotina de trabalho do usuário (na visão do projetista).

4.4.3.3 Pré-diagnóstico

Na fase de pré-diagnóstico é feita uma avaliação prévia das características do *software*, com base no conhecimento de ergonomia de interface e ergonomia cognitiva e na experiência do avaliador. Nessa fase são levantadas hipóteses de problemas mais comuns encontrados nessa primeira avaliação, para depois esses pontos serem testados, observados e comprovados nos testes de usabilidade. Para essa atividade, o avaliador pode fazer uma avaliação heurística ou se utilizar de *checklists* especialistas.

4.4.3.4 Definição dos cenários

Para a preparação dos ensaios é preciso ter claro quem serão os usuários e quais serão suas tarefas. Para isso é preciso reconhecer o perfil do usuário, elaborar questionários sobre o usuário e suas tarefas, definir tarefas que serão observadas e obter uma amostra de usuários aptos a efetuarem os ensaios.

4.4.3.4.1 Reconhecimento do perfil do usuário

Os projetistas, por já possuírem um conhecimento anterior das fases de construção do sistema, podem orientar o analista na escolha das pessoas que poderão fazer parte dos ensaios de interação, e quais são seus perfis, cabendo ao

analista fazer um reconhecimento mais detalhado dessas pessoas. Nessa fase de reconhecimento, o analista faz um primeiro contato com os usuários potenciais do *software*. Sugere-se uma visita ao local de aplicação dos ensaios, e uma solicitação de permissão da chefia para o envolvimento desses indivíduos na avaliação. Da mesma forma devem-se informar as pessoas sobre a finalidade da avaliação, quais os procedimentos tomados, qual será a participação delas, deixando-as livres para aceitarem, ou não, fazer parte dos ensaios. É preciso deixar claro que elas poderão abandonar os ensaios caso sintam-se constrangidas de alguma forma. Deve-se ter o consentimento dos envolvidos, e não somente a autorização da chefia.

4.4.3.4.2 Elaboração de questionários sobre o usuário e sua tarefa

Nesta fase é preciso obter dados mais detalhados a respeito dos usuários e suas tarefas. Esses dados poderão ser coletados através de questionários formais, ou com roteiros para uma entrevista conduzida. Podem-se coletar dados a respeito:

a) dos recursos disponíveis, tanto técnicos como físicos para a realização da tarefa; verificar qual o tipo de equipamento e versão do *software* que o usuário está utilizando; identificar problemas de estrutura física e organizacional que possam interferir na realização da tarefa; e identificar se o usuário possui treinamento e suporte por parte da empresa e da equipe técnica que desenvolveu o sistema;

b) da adaptação do contexto da tarefa ao usuário e seu relacionamento com outras atividades; observar o vocabulário do usuário, as pressões que sofre pela chefia, interferências dos colegas de trabalho, tempo de atividade, conhecimentos em informática, ou em outros sistemas da empresa, esses dados são necessários para identificar o grau de especialidade em informática, se podem ser caracterizados como novatos ou experientes;

c) da utilização do sistema; identificar as funcionalidades do sistema que o usuário julga críticas, de maior impacto, quais pontos positivos e negativos em relação ao seu trabalho e freqüência de utilização.

4.4.3.4.3 Definição de tarefas para os cenários

Para uma avaliação ergonômica é interessante que as tarefas sejam aquelas relacionadas com pontos conflitantes entre o que o analista observou no pré-diagnóstico e os indícios fornecidos pelos usuários através das entrevistas. Além de avaliar as diferentes funcionalidades do sistema, a construção dos cenários deve ter embutidos também objetivos do projetista para que esses também possam ser testados. Os cenários e as tarefas devem então ser bem definidos, pois os métodos de coletas possuem limitações, demandando de tempo e pessoal para sua realização.

4.4.3.4.4 Obtenção da amostra de usuários

É preciso reconhecer quem efetivamente realiza as tarefas que compõem os cenários da avaliação. Recomenda-se que sejam utilizados usuários de diferentes graus de experiência, definidos como novatos e experientes, pois os usuários novatos darão maiores informações sobre as facilidades de aprendizagem e a simplicidade de utilização. Já os experientes, darão mais informações sobre a organização das funções e a repartição das informações.

Já o tamanho da amostragem deve cobrir os diferentes tipos de usuários que possam utilizar o *software*. Reafirma-se a necessidade de deixar claro aos participantes os objetivos e a extensão da avaliação, qual o envolvimento e o que se espera deles.

4.4.3.5 Ensaios de interação

A boa preparação dos ensaios de interação determina fortemente os resultados das avaliações. Nesta etapa do processo estão os ajustes dos cenários, preparação dos ensaios de interação, realização dos ensaios, coleta e análise dos dados e diagnósticos e relatório final.

4.4.3.5.1 Ajustes dos cenários

Para cada usuário que participe dos ensaios, deve ser feita nova entrevista para adequar as tarefas ao seu grau de entendimento.

4.4.3.5.2 Preparação dos ensaios de interação

A preparação dos ensaios envolve preocupações com relação aos equipamentos que serão utilizados nos registros das interações; à escolha das técnicas de observações; à definição de estratégias de soluções de impasses e problemas ocasionais; e a preservação do anonimato do usuário, para que não causem problemas futuros.

Para diminuir situações de constrangimento dos usuários em momentos de impasse, o analista deve adotar a postura de deixar o usuário resolver sozinho qualquer tarefa. Caso ultrapasse alguns minutos em uma determinada situação de erro, deve-se conduzi-lo à solução do problema de forma educada, dando a entender o erro do *software* e não do usuário. Desta forma não o inibimos à continuação do ensaio. Pode-se ainda determinar uma outra tarefa prevista no ensaio, ou cancelar a sessão, caso note que o usuário estiver realmente incomodado, nervoso ou constrangido.

4.4.3.5.3 Realização dos ensaios

Os ensaios podem ser realizados tanto em laboratório como no ambiente de trabalho do usuário. A condução dos trabalhos deve ser definida pelo ergonomista, limitando-se também a quantidade de pessoas envolvidas na interação, como número de avaliadores, técnicos do equipamento de filmagem e um usuário.

4.4.3.5.4 Coleta, análise dos dados, diagnóstico e relatório final

A coleta de dados é feita através de gravações em vídeo das sessões de ensaios. Após as sessões, os analistas revêem em todas as gravações a busca de

dados relevantes que comprovem ou não as hipóteses formuladas no pré-diagnóstico. Outras situações que não foram previstas poderão ser encontradas nas gravações, já que as sessões são reproduções das reais situações de uso. A descrição e comentário da análise e o diagnóstico de problemas são entregues em forma de relatório aos projetistas, que poderão fazer uso desse material para reparos no *software*, na versão atual, ou em versões futuras.

4.4.4 Testes de uso ativo

Algumas formas de avaliação da interface em testes de uso ativo, ocorrem através de entrevistas e discussões em grupos de usuários; *log* de utilização; consultorias *on-line* ou por telefone; caixas de sugestões *on-line*, *newsgroups* e *newsletters*.

4.5 FORMAS DE REGISTRO DOS DADOS OBSERVADOS DURANTE AS AVALIAÇÕES

As técnicas e métodos de inspeções necessitam de registros dos dados que são gerados durante as avaliações, estas formas de registros devem ser utilizadas como auxiliares no processo de avaliação, possibilitando ao analista rever e recuperar dados não percebidos durante as avaliações.

Essas tecnologias de apoio são os recursos de registro como gravação em audio e em vídeo, utilização paralela de gravações através de softwares espiões instalados no computador durante a execução das tarefas, anotação em papel e caneta, transcrição, esquemas de códigos, e utilização de protocolos verbais concorrentes ou vervalizações consecutivas em sessões de post-task walkthroughs. Estas tecnologias de apoio estão descritas a seguir.

4.5.1 Papel e caneta

O uso de anotação simples através de registros feitos pelo observador com

papel e caneta exige grande habilidade por parte do observador, apesar de ter uma conotação *low tech*. Essa técnica não deve ser descartada nem subestimada, é bastante trabalhosa e produz uma descrição de alto nível da tarefa.

Após um pequeno número de observações piloto, é usualmente possível o observador criar um esquema de códigos para reduzir a quantidade e tempo devotado na escrita, restando mais tempo para observar a tarefa e encorajando o observador a estabelecer o nível de detalhe e selecionar o que observar, reduzindo o estresse e o esforço gasto na observação.

A vantagem de possuir um esquema de códigos, além da escrita mais fácil, é a de auxiliar na posterior análise desses dados.

Papel e caneta são portáteis e podem ser usados em uma grande variedade de ambientes, porém limitam-se ao nível de detalhe que se pode gerar. Durante o uso do método papel e caneta, é possível adotar o uso de protocolos verbais concorrentes para auxiliar o processo de registro, pois enquanto o usuário descreve sua atividade, retarda o desempenho da tarefa, possibilitando o registro por parte do observador.

4.5.2 Gravação em áudio

É raro o uso de gravação em áudio sozinho. Geralmente opta-se por recurso como complemento do método papel e caneta como fonte adicional de dados na sua análise posterior, transcrição e complementação de informações. Essa técnica reduz consideravelmente o trabalho do observador durante a realização dos ensaios.

4.5.3 Gravação em vídeo

É uma tecnologia bastante popular e conhecida. As gravações podem ser feitas no próprio local de trabalho do usuário ou em laboratórios.

No caso da opção em se efetuar a gravação no local de trabalho, mantém-

se mais a fidelidade, uma vez que é possível a reprodução de situações reais de uso, em que ocorrem interferências do ambiente de trabalho, como interrupções pelo telefone, pelos colegas de trabalho, pressão por parte da chefia, etc.

Gravações em laboratório consistem de um ambiente neutro, que se constitui, geralmente, de dois ambientes, um onde o usuário irá permanecer juntamente com os equipamentos de uso e de gravação, e outro separado por vidros espelhados, onde ficam os observadores.

Optando-se pelo uso de somente uma câmera de vídeo, tanto no local de trabalho como em laboratório, o ideal é o posicionamento da câmera atrás do usuário, sobre o ombro, focando apenas a tela do computador. É interessante também o uso de um relógio temporizador durante as gravações, para registrar o tempo gasto em cada observação.

4.5.4 Protocolos verbais concorrentes

Também conhecida como verbalização simultânea, em que o analista, ao observar a tarefa, solicita ao usuário que vá comentando as ações que está tomando. Dessa forma é possível entender os objetivos do usuário, e o porquê de ter executado determinada ação. São usadas perguntas do tipo: o que está tentando fazer?; o que está lendo?

Esse tipo de verbalização fornece informações importantes ao observador, mas pode interferir na qualidade do desempenho da tarefa, já que muitas pessoas consideram uma carga bastante alta o fato de explicar enquanto estão trabalhando. Pessoas muito extrovertidas também poderão desviar a atenção do trabalho para explicar excessivamente sua atividade. Cabe ao observador dosar a quantidade de verbalização durante as sessões e tentar não interferir no trabalho do usuário.

4.5.5 Verbalização consecutiva ou *Post-task Walkthroughs*

É uma técnica que é aplicada após a sessão de interação, sendo solicitado ao usuário que comente a forma como conduziu o trabalho, que explique as ações que tomou em determinado momento. Caso o usuário se esqueça de algum detalhe, poderá recorrer às gravações em vídeo para que comente o que aconteceu. Outra

forma seria solicitar ao usuário que comente certas características da interface, dessa forma pode-se obter informações e sugestões sobre pontos positivos ou negativos do usuário em relação ao *software* (CYBIS, 1997).

Após a observação da tarefa, independente da tecnologia adotada para seu registro, esta deve gerar um texto das atividades realizadas. Dessa forma os registros devem ser decodificados de forma a serem por duas técnicas: transcrição e esquema de códigos.

De posse das gravações em vídeo ou fitas de áudio, essas devem ser transcritas de forma a gerar listas de atividades. Esse trabalho requer muita paciência, já que é uma atividade morosa, pois muitas vezes é preciso recorrer a usos de congelamento e retrocesso da fita, para entender melhor determinadas situações e, então sim, obter dados significativos.

Esquema de códigos é uma técnica manual consistente, coerente e rápida para registrar a observação da tarefa. Informa ainda que essa técnica pode ser utilizada tanto na transcrição de vídeos como nas observações diretas com usuários.

Ramos (1996) recomenda ainda que é necessário decidir qual o nível de detalhe dos registros que deverão ser codificados, que para a geração de tal código sâo recomendáveis algumas transcrições piloto, e que o código não deve ser rígido, podendo ser incluídas nele novas categorias, caso sinta-se a necessidade.

4.6 CONCLUSÃO COM RELAÇÃO AOS MÉTODOS DE INSPEÇÕES

Métodos de inspeção diferentes possuem metas ligeiramente diferentes, mas todos têm embutido um “intencionado” caminho para avaliar interfaces e encontrar problemas de usabilidade. Estas avaliações são baseadas no julgamento do inspetor. Os métodos de inspeção individuais variam de acordo com o julgamento de cada avaliador. E é na confiança nesses julgamentos que acontece a realimentação da avaliação de elementos específicos de uma interface.

O conhecimento ergonômico e a experiência dos avaliadores, assim como a

apreciação prévia do contexto de uso do sistema, são fatores significativos para o sucesso da avaliação por meio de métodos de inspeção.

A participação do usuário na avaliação e nas demais fases de desenvolvimento do sistema traz como benefício, além do comprometimento do usuário, a sua satisfação, uma vez que ele percebe que seus anseios estão sendo levados em conta para melhorias da ferramenta que utiliza.

Conforme Moço (1996, p. 39) “(...) as técnicas que não utilizam usuários na sua interação, não deixam de ser válidas quando utilizadas como uma ferramenta de apoio para o pré-diagnóstico, do qual se traçam as hipóteses para serem validadas ou não”.

Pesquisa efetuada por Jeffries (1991), em estudo de caso para comparação da eficiência de técnicas de avaliação heurística, testes de usabilidade, *checklists (guideline*s) e navegação cognitiva para “encontrar tudo o que afetasse a utilização”, resultou nas seguintes conclusões:

QUADRO 2: MÉTODOS DE AVALIAÇÃO DAS VANTAGENS E DESVANTAGENS

MÉTODO	***VANTAGENS***	***DESVANTAGENS***
Avaliação heurística	Identifica maior número de problemas Identifica problema sérios Possu baixo custo	Requer experiência em ergonomia Requer muitos avaliadores
Testes de usabilidade	Identifica problemas sérios e problemas repetidos Ignora problemas de baixa prioridade	Requer experiência em ergonomia Tem custo alto Ignora problemas de consistência
Guidelines (*checklists*)	Identifica erros repetitivos e gerais Pode ser usado por desenvolvedores de *software*	Ignora problemas alguns problemas severos
Cognitive walk-through (navegação cognitiva)	Ajuda na definicao de objetivos de usuários e seus anceios Pode ser usado por desenvolvedores	Necessita de definição de metodologia É tedioso Ignora problemas gerais e repetitivos

FONTE: JEFFRIES (1991, 10)

5 DESCRIÇÃO DO ESTUDO DE CASO PARA APLICAÇÃO DO MÉTODO E APRESENTAÇÃO DOS RESULTADOS E ANÁLISE

Após estudo de categorias de métodos encontrados na literatura, o método utilizado por Ramos (1996) foi classificado como sendo do tipo analítico empírico-exploratório, tendo sua aplicação e teste por meio de estudo de caso.

Escolheu-se para aplicação do método de avaliação um *software* de comercialização agro-ecológica da Associação de Agricultores Ecológicos da Encosta da Serra Geral (AGRECO), situada na Cidade de Santa Rosa de Lima, em Santa Catarina. Esse sistema faz parte de um projeto de informatização denominado AgroREDE, que tem parceria com a Universidade Federal de Santa Catarina, através da qual estudantes de mestrado participam e contribuem com estudos complementares e auxiliares desse projeto.

Partiu-se da premissa de que *softwares* muitas vezes são desenvolvidos às pressas e privilegiam mais a lógica de programação do que as facilidades de uso. No caso do *software* da Agreco, detectam-se indícios desse problema, uma vez que foi observado que algumas necessidades do usuário não foram atendidas, mas elaboradas de outra forma devido às facilidades de programação.

A facilidade de utilização é fator importante para o desempenho das tarefas de comercialização no *software* da Agreco. Justifica-se essa preocupação, visto a peculiaridade que este apresenta. Sendo um aplicativo para uso da comercialização agrícola, para uma associação de produtores, os usuários do aplicativo são pessoas ligadas ao processo de produção rural, com baixa escolaridade. A sede da Associação Agreco localiza-se em uma cidade de pequena população e afastada de

grandes centros, o que a torna escassa de mão-de-obra especializada em informática. As pessoas que trabalham na AGRECO possuem um grau de especialidade em informática bastante baixo, identificados como usuários iniciantes. Porém, os que lidam com o aplicativo possuem grande domínio do processo de comercialização em si. Dentro desse quadro, torna-se importante que o *software* tenha a interface ergonômica.

5.1 ESCOLHA DE MÉTODOS DE APOIO AO MÉTODO UTILIZADO POR RAMOS

O método de Ramos (1996) em sua primeira etapa indica a transcrição de todas as sessões gravadas em fita com concomitante sinalização das situações problemas. Está implícito que o método compreende um estudo prévio de análise do *software* e de preparação, planejamento e efetivação de uma observação "*in loco*" da tarefa informatizada.

Dessa forma, juntamente com a aplicação do método utilizado por Ramos (1996), foram empregadas técnicas preditivas, mais explicitamente, os métodos de avaliação heurística para avaliação preliminar do *software*, com apoio de *checklists* e recomendações ergonômicas de Scapin e Bastien (1997), e ensaios de interação, que contaram com a colaboração de usuários, em situações reais de uso.

As avaliações heurísticas foram executadas no Laboratório Sistemas de Conhecimento (LSC) da UFSC. Já os ensaios de interação foram realizados na própria sede da Agreco, na cidade de Santa Rosa de Lima – SC, utilizando-se de todos os usuários do sistema. Sendo eles de número pequeno, procurou-se repetir algumas interações para diagnosticar problemas ergonômicos. Outros funcionários da sede, que não são usuários diretos do *software*, também foram solicitados a participar dos ensaios, por serem pessoas envolvidas e profundos conhecedores do processo. Optou-se por esse recurso para verificar a eficiência da interface em situações que o usuário "oficial" se ausentasse, diagnosticando assim se o componente conhecedor do processo conseguiria executar a tarefa com o uso do aplicativo.

Com base na bibliografia estudada, comprovou-se a eficiênica e os bons resultados obtidos através dos ensaios de interação. Outro fator que auxiliou na

decisão pelos métodos aplicados como complementares ao método de Ramos (1996) foi o trabalho relatado por Nielsen (2002), que orienta a aplicação de avaliação heurística juntamente com ensaios de interação.

Sendo assim, sugere-se a inclusão de dois passos (1A e 1B) anteriores a primeira etapa do método proposto por Ramos (1996), que orientem claramente o usuário do método como proceder uma avaliação prévia para conhecimento e diagnóstico do *software* a priori. Em seguida, sugere-se a inclusão de passos que orientem para a preparação da atividade de observação da tarefa em si, como preparação de cenários, equipamentos a serem usados, etc, passos aqueles sugeridos por Cybis (1997) descritos no Capítulo 4, Tópico 4.4.3, *Ensaios de Interação.* Os passos adicionados podem ser descritos da seguinte forma:

1A : fazer avaliação prévia do sistema informatizado, por meio de aplicação de avaliação heurística e inspeções cognitivas (w*alkthrough cognitive*), com apoio e *checklists* e/ou recomendações ergonômicas. Sugere-se a utilização de recomendações ergonômicas de Scapin e Bastien (1997), por este já ser avaliado em outras aplicações e ter se demonstrado bastante detalhado e estruturado. Outra sugestão é a utilização de *checklist* disponível na página do Laboratório de Utilizabilidade, no endereço ergolist (http://www.labiutil.br-ergolist/). Este fornece sequências e explicações bastante pertinentes a uma avaliação heurística executada por especialistas em usabilidade ou não.

1B: preparação dos ensaios de interação, apoiados pelas recomendações de Cybis (1997) detalhadas no Capítulo 4.

5.2 ALTERNANDO APLICAÇÃO DE AVALIAÇÃO HEURÍSTICA E TESTES COM USUÁRIOS

Embora a avaliação heurística encontre muitos problemas que não são encontrados em testes com o usuário, o contrário também é verdadeiro, ou seja, é possível que se encontrem erros em testes com usuários que não seriam encontrados em avaliações heurísticas.

É provável que avaliadores aplicando somente avaliação heurística negligenciem alguns problemas de usabilidade se o sistema for altamente dependente do domínio e os avaliadores não tiverem habilidades no domínio.

Em alguns estudos de caso relatados por Nielsen (2002) os problemas que foram encontrados eram tão específicos do domínio que não teriam sido encontrados se não fossem os testes com usuários.

Desde que avaliação heurística e testes com usuários encontram erros diferentes, a recomendação é que os métodos sejam usados em conjunto.

O analista deve executar uma avaliação heurística primeiro para “limpar” a interface e eliminar tanto quanto possível os problemas mais óbvios. Depois de um redesenho da interface, então é feito o teste com usuários para conferir a interface com o *design* e encontrar problemas restantes que não haviam sido diagnosticados pela avaliação heurística.

Há duas razões para alternar entre avaliação heurística e testes com usuários:

a) a passagem de avaliação heurística pode eliminar vários problemas de usabilidade sem a necessidade de “desperdiçar” usuário, por ser, às vezes, difícil ter usuários disponíveis para testes, por estarem em outros locais ou por serem em grande número;

b) essas duas categorias de usabilidade mostram conjuntos distintos de problemas encontrados, então elas se completam, ao invés de conduzir a achados repetidos (DESURVIRE et al., 1992 ; JEFFRIES et al., 1991 ; KARAT et al., 1992 apud NIELSEN, 2002).

No caso do sistema da Agreco, os usuários estavam em uma localidade distante, e as visitas para aplicação de ensaios de interação tiveram que ser muito bem planejadas, para evitar novas viagens até o local onde se encontravam os

usuários alvo do sistema. Isso vai ao encontro da sugestão do autor, que também aconselha que, caso “(...) os usuários do sistema estejam dispersos é oneroso o teste com usuários, então sugere-se o uso de avaliações heurísticas e testes com usuários em laboratórios, testando um número maiores de repetição para polir a interface antes do produto ser lançado no mercado” (NIELSEN, 2002).

5.3 DESCRIÇÃO DOS RESULTADOS DA FASE PRELIMINAR: AVALIAÇÃO HEURÍSTICA

A avaliação heurística foi efetuada conforme relatado acima. Num primeiro contato com o aplicativo, foram feitas inspeções rigorosas de cada tela, apoiadas por critérios ergonômicos de Scapin e Bastien (1997) para nortear as inspeções. Essa inspeção prévia, apoiou-se também na exploração cognitiva, tentando-se reproduzir os pensamentos e ações de usuários leigos, que estivessem se confrontando pela primeira vez com a interface.

À medida que eram diagnosticados problemas com a interface, estes eram anotados e justificados, embasados nos critérios ergonômicos de Scapin e Bastien (1997), e em seguida esses relatórios eram encaminhados para a equipe de projeto e programação para decisão quanto às correções das falhas. Nesse processo, notou-se a necessidade de criar uma forma de representação que facilitasse a comunicação e o entendimento entre o avaliador e a equipe técnica do projeto. Para isso, foi criada uma simbologia que identificasse e classificasse os tipos de problemas encontrados e seu grau de prioridade para correção. Isso se fez necessário, vista a rapidez do processo de mudança, o amadurecimento do produto e o seu prazo de entrega.

5.3.1 Criação de simbologia e atribuição para graus de severidade:

Com base nas indicações de sinalização de prioridades em problemas diagnosticados, Nielsen (2002) sugere o uso de graus de severidades, em que a avaliação de severidade pode ser usada para alocar mais recursos, para eliminar

problemas mais sérios, e pode prover também estimativa de necessidade de esforços adicionais de usabilidade.

Pode-se decidir lançar um produto se os problemas encontrados e julgados forem considerados de natureza apenas cosmética.

A severidade de um problema é a combinação de três fatores:

a) a freqüência em que ele ocorre: se é comum, ou raro;

b) o impacto do problema, caso ele ocorra: se é fácil ou difícil para o usuário superá-lo;

c) a persistência do problema: se ele é um problema único, que uma vez o usuário conhecendo poderá superá-lo, ou se o usuário é repetidamente aborrecido por ele;

É preciso avaliar o impacto do problema para o tipo de usuários-alvo do sistema, tendo claro que alguns problemas de usabilidade têm efeito devastador na popularidade do produto.

Nielsen (2002) sugere o uso de taxas de escala de 0 a 4, apresentadas abaixo:

0 = não concordo, esse não é um problema de usabilidade;
1 = problema somente cosmético;
2 = problema de usabilidade secundário, prioridade baixa;
3 = problema de usabilidade principal, devendo dar prioridade alta;
4 = catástrofe de usabilidade, imperativo solucionar o problema antes do seu lançamento.

Porém, optou-se por usar de simbologia, utilizando recursos gráficos para representar graus de severidade. Definiu-se os seguintes símbolos e significados:

☹ Problemas de funcionamento (erros/ou necessitam de reparos urgentes);

😐 Sugestões de melhorias/facilidades de uso (depende de definições com os responsáveis do projeto);

☺ Recomendações ergonômicas (versões futuras/ alteração de *lay-out*/melhorias) (depende de definições com os responsáveis do projeto).

A avaliação de severidade deve ser aplicada após as sessões de avaliação. Do contrário poderá atrapalhar o avaliador, e ele será seduzido a tentar classificar o erro, perdendo o foco da busca por novos problemas de usabilidade. Nesse caso, recomenda-se uma avaliação inicial, para só então se fazer a taxação de graus de severidade de cada problema encontrado.

Na aplicação de severidade é interessante que vários avaliadores atribuam os graus de severidade, pois o trabalho cooperativo tende a ser muito mais proveitoso. O ideal, conforme sugere a literatura, seria o uso de 3 inspetores para uso da avaliação heurística (NIELSEN, 2002). Pois, considerando que cada avaliador encontra um conjunto de problemas e atribui a eles seu grau de severidade, esses podem ser depois unidos com os conjuntos dos demais avaliadores, e então juntos poderão dar prioridades consensuais aos problemas encontrados. É possível que alguns erros não tenham sido encontrados pelos demais avaliadores, então o ideal seria que cada avaliador, desse um grau de severidade para esses problemas, para depois serem discutidos em conjunto com os outros avaliadores.

Nesta dissertação a avaliação heurística propriamente dita foi efetivada por um único indivíduo. Porém, se acredita que esse fator tenha sido minimizado, uma vez que os relatórios eram entregues na maioria das vezes pessoalmente, e os resultados eram discutidos com os projetistas e programadores, quando eram revistos os pontos falhos na própria interface do *software*, explicando os problemas encontrados. Essa atividade pode ser considerada uma espécie de trabalho colaborativo, conforme a sugestão do autor, estando implicitamente atendida a recomendação do autor em se utilizar no mínimo 3 avaliadores (NIELSEN, 2002). Um exemplo de relatório entregue à equipe técnica com atribuições de graus de severidade pode ser observado no Anexo 2 desta dissertação.

Após essa análise preliminar do *software*, partiu-se para a preparação os ensaios de interação.

5.4 OS ENSAIOS DE INTERAÇÃO

Os ensaios de interação foram preparados e aplicados seguindo orientações de Cybis (1997)

A definição do cenário compreende as etapas de reconhecimento do perfil do usuário, elaboração do questionário, coleta de dados, definição das tarefas para o ensaio, obtenção dos usuários para os ensaios, definição do público alvo, preparação dos ensaios, gravação e protocolos verbais.

Foram idenficados os usuários potenciais do sistema. Visto que o número de usuários era pequeno, identificou-se através de entrevistas o grau de especialidade com relação ao uso de ferramentas computadorizadas. Foram identificadas nas categorias um usuário que utilizava o sistema com mais domínio, um outro iniciante em informática com profundo conhecimento do processo manual, e um terceiro iniciante na informática e no processo.

O módulo avaliado foi o do sistema de comercialização da Agreco, mais especificamente os processo de lançamento de pedidos, estimativas de colheita e geração de notas fiscais, por serem os principais em importância e frequência de uso.

5.4.1 Ensaios

Os ensaios compreendem as etapas de ajustes dos cenários, determinação do número de usuários, preparação dos ensaios de interação (tempos), equipamentos utilizados para avaliação (câmeras, computadores), realização dos ensaios, estratégias para situações de constrangimento, coleta e análise de dados.

Na primeira sessão de ensaios deixou-se livre a utilização do *software*. O aplicativo passava por constantes mudanças e ajustes. Um *software* já existente estava sendo substituído pelo que está sendo avaliado. Então, possíveis vícios do sistema antigo poderiam ser detectados na utilização do *software* que estava sendo instalado.

Em seguida foi solicitado que os usuários efetuassem tarefas corriqueiras do sistema, como lançar pedidos, lançar estimativas, editar colheita efetiva, gerar estimativa de colheita e emitir nota fiscal.

Segundo sugestão de Nielsen (2002), optou-se por repetir as sessões de ensaio para se obter mais informações sobre problemas da interface, uma vez que o número de usuários disponíveis era pequeno.

Os ensaios foram gravados em fitas VHS, utilizando verbalização simultânea e o usuário foi previamente instruído sobre as gravações e objetivos dos ensaios. O usuário participante da pesquisa foi informado que poderia interromper a sessão a qualquer momento caso se sentisse constrangido, ou de alguma forma estivesse desgostoso. O sigilo da identidade foi preservado nos relatos aqui descritos. Foram utilizados protocolos verbais concorrentes, conforme sugerido por Ramos (1996), e solicitado ao usuário que narrasse as atividades que estava desempenhado, e os objetivos que pretendia alcançar em cada ação tomada. Adotou-se a postura de observar e anotar falhas importantes durante o processo de observação e, caso ocorresse algum impasse, passados alguns minutos, o avaliador interferiria perguntando e tentando ajudá-lo a resolver aquela situação de embaraço.

A filmadora foi posicionada atrás do usuário na altura do ombro, focalizando somente o monitor. Não foram utilizados outros recursos, como *softwares* espiões, para auxiliar na coleta de dados.

Após os ensaios de interação, foi aplicado então o método utilizado por Ramos (1996) que será agora relatado através de passos numerados para melhor entendimento durante a aplicação do método:

1 - Transcrição de todas as sessões gravadas em fitas com concomitante sinalização de situações problemas;

2 - A partir dos dados oriundos da observação da tarefa, listagem de todos os objetos e ações específicas encontradas;

3 - Descrição genérica para a tarefa;

4 - Descrição hierárquica das ações e objetos componentes da tarefa (usando diagramas propostos por Diaper (1989));

5 - Descrição de todas as operações (unidades procedurais ou ações específicas na denominação proposta por Diaper (1989))

6 - Definição, segundo o modelo de Barthet (1988), de um plano geral de

ações para a tarefa, ou seja, obter uma visão macro do sincronismo das ações presentes na tarefa;

7 - Realização da avaliação ergonômica da aplicação.

A cronologia da realização dos passos não seguiu exatamente conforme o numerado. A parte de transcrição das fitas (passo 1) é uma tarefa demorada e trabalhosa, que visa identificar falhas que não foram percebidas no ato da observação. As gravações também são úteis no caso de rever com os usuários situações de empasse, nos quais ele poderá então explicar o porquê de ter tomado determinadas ações.

Um exemplo de transcrição de uma sessão é demonstrada no Anexo 1, em que utilizou-se da simbologia oferecida por Ramos (1996), sendo alguns símbolos alterados e incorporados na medida em que se fizeram necessários.

Os passos 2, 3 e 4 foram executados em paralelo, sendo que eles se apoiam mutuamente. A descrição genérica de alto nível para a tarefa (passo 3) foi definida como “lançar pedidos”. A construção hierárquica das ações e objetos componentes da tarefa (passo 4) é relatada abaixo.

5.4.2 Descrição dos objetos e ações que compõem a tarefa (passo 4)

/
/ **----Natureza da ação**
/ | **----- editar (conferir/ salvar/cadastrar)**
/ | **----- consultar (ler/cancelar)**
/ | **----- imprimir (classificar)**
/ | **----- gerar**
/----Objeto genérico
| **----- Cadastro**
| **----- Colheita**
| **----- Pedidos**
| **----- Notas fiscais**
| **----- Linhas**

5.4.3 Descrição hierárquica dos objetos do ambiente do sistema de comercialização da Agreco

|------Cadastro
| | ---------- Produtos
| | ||----------Dados cadastrais
| | ||----------Categoria dos produtos
| | ---------- Cliente
| | ||----------Dados cadastrais
| | ||----------Categoria de nota fiscais
| | ||----------Rapel
| | ||----------Percentuais de descontos
| | ---------- Condomínio
| | ||----------Associados
| | ||----------Dados cadastrais
| | ||----------Produtos produzidos
| | ||----------Despesas
| | ---------- Preços
| | ||----------Categoria de preços
| | ||----------Tabela de preços
| | ||----------Preços por cliente
| | ||----------Preços por categoria
| | ||----------Promoções
| | ---------- Banco
| | ||----------Nome
| | ||----------Agência
| | ---------- Despesas
| | ||----------Descrição
| | ||----------Código
| | ---------- Cidades
| | ||----------Cidade
| | ||----------Estado
| | ---------- Transportadores
| ||----------Dados cadastrais
| ||----------Clientes (atendidos)
| ||----------Linhas
| ----- Colheita
| ||---------- Estimativa
| ||---------- Efetiva
| ||---------- Encomenda
| ----- Nota Fiscal
| ||---------- Ajuste/impressão
| ||---------- Cancelamento/re-impressão
| ||---------- Nova nota
| ------ Pedidos
| ||--------- Do mercado
| ||--------- Cestas
| ||--------- Por produto
| ||--------- Por condomínio
| ||--------- Ajustes
| ----- Cotas

| ||-------- Previstas
| ----- Linhas

5.4.4 Descrição hierárquica das ações da tarefa “lançar pedido”

| ----- Editar(lançar/cadastrar)
| | ----------Condomínio
| | | ----------Dados cadastrais
| | | ----------Produtos produzidos
| | | ----------Despesas
| | ----------Banco
| | | ----------Nome
| | | ----------Agência
| | ----------Despesas
| | | ----------Descrição
| | | ----------Código
| | ---------- Cliente
| | | ----------Dados cadastrais
| | | ----------Categoria da nota fiscal
| | | ----------Rapel
| | | ----------Percentuais de descontos
| | |----------Pedidos
| | |----------Quantidades de produtos
| | |----------Trocas
| | | ---------Ajustes
| | ----------Preços
| | |----------Categoria de preços
| | |----------Tabela de preços
| | |----------Preços por cliente
| | |----------Preços por categoria
| | ----------Cidades
| | |----------Cidade
| | |----------Estado
| |----------Transportadores
| | |----------Dados cadastrais
| | |---------- Clientes (atendidos)
| | ---------- Linhas
| |----------Colheita Efeitva
| | |----------quantidade de produtos
| |----------Nota Fiscal
| | | ----------Número de nota fiscal
| | | ----------Categoria de nota fiscal
| |----------Estimativas
| -------Consultar (conferir)
| | ----------Pedidos
| |----------Clientes
| |
|--------Classificar (NF)
| |----------Nota fiscal
|------- Salvar

| |----------Pedidos
|------- Cancelar
| |----------Pedido
| | ----------Nota fiscal
|-------- Imprimir
| |----------Nota fiscal
| |----------Estimativa de colheita
| |----------clientes
| |----------linhas
|-------- Gerar
** |----------estimativa de colheita**

5.4.5 Descrição hierárquica dos objetos e ações

O passo 5 do método de Ramos (1996), descrição das operações não foi efetivado, levando em consideração a dinâmica no processo de mudanças que o sistema sofria durante esta avaliação. Esse passo é bastante útil no caso de gerar um roteiro detalhado de cada operação do sistema, porém ele foi excluído nessa aplicação por ter sido considerado demasiado trabalhoso e infrutífero, uma vez que as contribuições nas avaliações anteriores já tinham sido agregadas pela equipe técnica, e a descrição de tais operações estavam sendo alterada constantemente.

O plano geral de ações da tarefa é uma maneira de representar graficamente a realização da tarefa (passo 6). Esta inclui condições de precedência, concatenamento, paralelismo e obrigatoriedade da execução de uma ação. No diagrama, a procedência obrigatória está indicada da esquerda para a direita, o paralelismo está na dimensão vertical. Ações que uma vez realizadas têm valor permanente durante a execução do sistema; são representadas por tracejamento descontínuo e podem ser desencadeadas a qualquer momento durante uma sessão de trabalho (RAMOS, 1996). A descrição das operações é demonstrada a seguir.

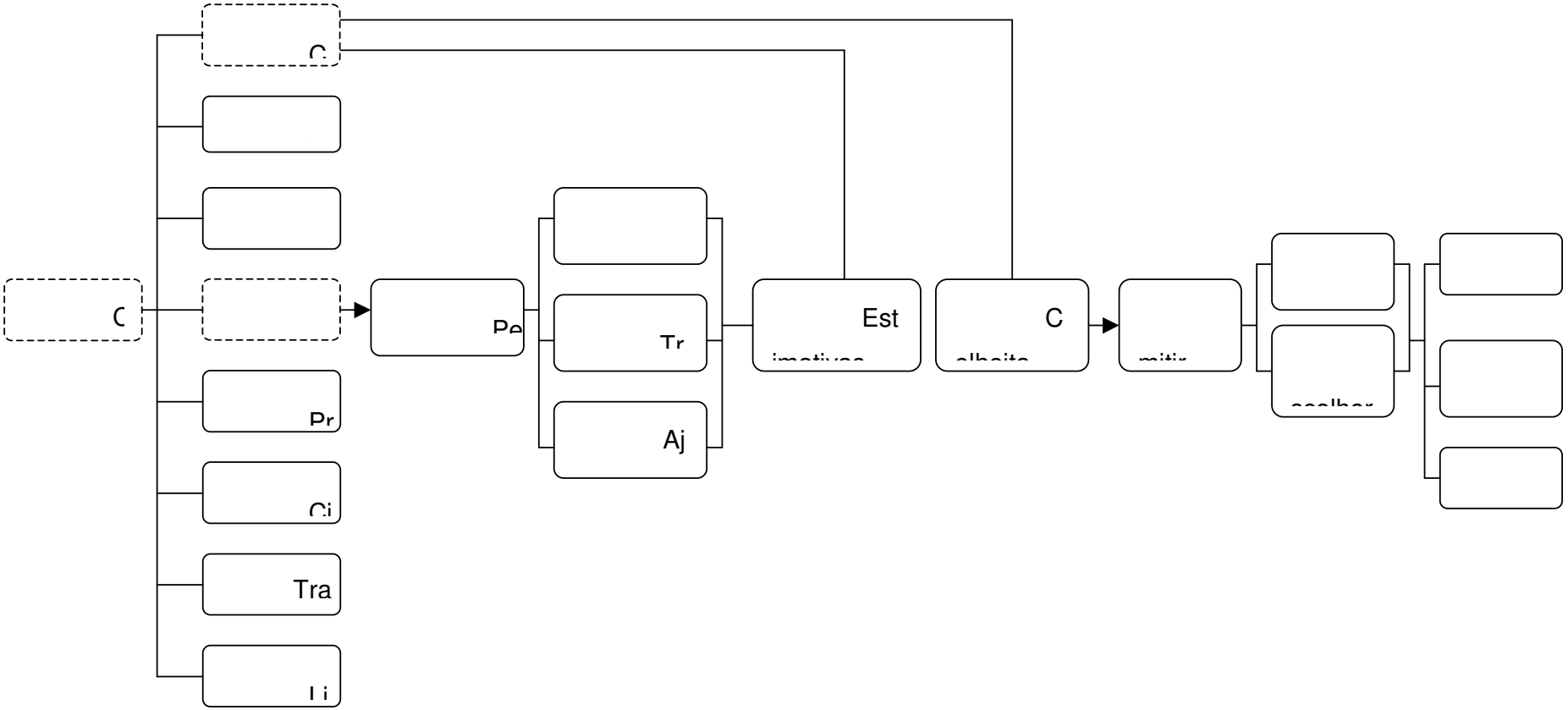


Figura 7: DIAGRAMA DE SEQUÊNCIA DE AÇÕES (Passo 6)

O sétimo e último passo, que compreende a avaliação ergonômica detalhada do *software*, não pôde ser aplicado. Isso porque o sistema estava em fase de implantação, com correções e ajustes contínuos, que não permitiram o seu desenvolvimento. Entretanto, na fase das avaliações heurísticas, foram levantadas recomendações ergonômicas, que encaminhadas à equipe técnica, contribuiram na qualidade ergonômica do *software*.

CONCLUSÃO

Na aplicação de um método de avaliação ergonômica, conseguem-se bons resultados, como dados confiáveis da situação ergonômica do *software*, e redução do tempo de aplicação. Desde que o método seja adequado, simples e objetivo, o que facilita também a sua execução.

No que diz respeito à avaliação ora apresentada, conclui-se primeiramente que se trata de um método não adequado para tal aplicação. Por se tratar de um *software* em implantação, o processo de alteração dos módulos era muito dinâmico, e o método não acompanhava o ritmo das modificações.

Esse método é adequado para outras situações, como a de sistemas que tenham sido construídos de modo artesanal e assistemático. Neste caso possibilita-se a produção de materiais e documentação detalhada do sistema. Pode ser utilizado também em fases iniciais do desenvolvimento, na análise e modelagem, excluindo-se a última etapa do método.

A outra premissa para avaliação do método é a simplicidade e objetividade, e pode-se afirmar que o método em estudo é muito trabalhoso, na transcrição dos dados das gravações, e na descrição de todas as operações identificadas na tarefa, como foi relatado no Capítulo 5.

O relatório completo contendo o resultado da avaliação ergonômica do *software* não foi efetivado na sua totalidade como propunha o método utilizado por Ramos (1996), pelos seguintes motivos: desde o início da aplicação do método, na utilização da avaliação heurística (relatados no Anexo 1), já eram gerados relatórios intermediários, contendo recomendações ergonômicas de melhorias ao aplicativo. O

software em estudo é muito peculiar à AGRECO, que passava por re-estruturação. Então, à medida que os módulos estavam sendo implantados, estes geravam questionamentos com relação aos procedimentos das atividades, desencadeando ajustes na organização administrativa da cooperativa, e alterações no sistema e em sua interface. Essa peculiaridade trouxe aprendizado tanto à organização quanto à equipe técnica do projeto, uma vez que houve um trabalho conjunto e cooperativo na construção dessa nova visão do funcionamento da organização.

Essa característica, por um lado, proporcionou a contribuição direta na avaliação do método dessa pesquisa, e, por outro, desfavoreceu o sequenciamento do trabalho de avaliação por parte do analista. Afinal, muitas vezes, os módulos avaliados deixavam de ser utilizados, ou eram alterados os procedimentos das atividades. O que faz concluir que essa metodologia limita-se à aplicação em sistemas mais sedimentados.

A combinação de avaliações heurísticas e testes de usabilidade agregados ao método original de Ramos (1996) mostrou-se eficiente, já que a avaliação heurística consegue identificar boa parte dos erros existentes, e é complementar na busca de erros por meio de ensaios de interação.

O último passo do método original não foi aplicado nesse estudo de caso, porém, é uma etapa importante. Ele deixa livre a escolha de critérios e recomendações ergonômicos, que possam ser utilizados como auxiliares. Essa pesquisa sugere o uso dos critérios e recomendações ergonômicas oferecidas por Scapin e Bastien (1997), por apresentarem um nível de detalhamento e hierarquização que facilitam a sua utilização.

O método utilizado nesta dissertação foi pouco explorado e aplicado até o momento. Sendo assim, as sugestões aqui agregadas são merecedoras de novos estudos.

REFERÊNCIAS BIBLIOGRÁFICAS

BARTHET, Marie F. **Logiciels intractifs et ergonomie**. Dunod. Paris, 1988.

CARVALHO et al. **Uma Estratégia para Implantação de uma Gerencia de Requisitos Visando a Melhoria Dos Processos de Software** Disponível em: <http://www.inf.puc-rio.br/~wer01/Pro-Req-3.pdf> Acesso em: 12 ago. 2002.

CYBIS, W. A. Desarmando armadilhas da informática. **Revista Inovar**, ago. 1998. n.13, p.13-15.

CYBIS, W. A. **Qualidades Ergonômicas.** Disponível em: <http://www.labiutil.inf.ufsc.br/qualidades.html> Acesso em: 02 mar.2001.

CYBIS. Walter de Abreu. **Abordagem ergonômica para IHC**. LabUtil-Laboratório de Utilizbilidade. Apostila de aula, 1997.

DIAPER Dan. **Scenarios and Task Analysis**. Interacting with Computers. 2002.

DIAPER, Dan ; ADDISON, Mark. **Task Analysis and systems analysis for software development:** Interacting with computers. v. 4, n. 1, 1992, p.124-139.

DIAPER, Dan. **Analysis focused Interview Data with Task Analysis for Knowledge Descriptions (TAKD).** Human-Computer Interaction, INTERACT'90, p. 227-282.

DIAPER, Dan. **Requirements Engineering:** Integrating HCI and Software Engineering Requirements Analysis: A Demonstration of Task Analysis Supporting Entity Modeling. Disponível em: <http://www.acm.org/sigchi/bulletin/ 1997.1/diaper.html> Acesso em: 23 jul. 2001.

DIAPER, Dan. **Task Analysis for Knowledge Description (TAKD):** The method and na example. In "Task analysis for human-computer interaction" edited by Dan DIAPER. Ellis Horwood Limited Publishers and John Wiley & Sons. New York, 1989. p.108-159.

DIAPER, Dan. Task Analysis for Knowledge Descriptions (TAKD): A Requiem for a

Method. **Behaviour and Information Technology**. v. 3, n. 20, 2001, p. 199-212.

DIAPER, Dan. **Task observation form human-computer interaction**. In "Task analysis for human-computer interaction" edited by Dan DIAPER. Ellis Horwood Limited Publishers and John Wiley & Sons. New York, c. 7, 1989.

DIAS, Claúdia. **Avaliação de usabilidade: conceitos e métodos**. Disponível em: <http://www.ii.puc_campinas.br/revista_ii/Segunda_edicao/Artigo_02/Avaliacao_de_usabilidade.pdf> Acesso em: 12 ago. 2002.

DIX, Alan et al. **Human-Computer Interaction**. Prentice Hall, 1993.

GONÇALVES, Daniel J. **Fatores Humanos:** Perfil do utilizador e modelos conceptuais. Disponível em: <http://mega.ist.utl.pt/~ic-ihm/programa/index.html> Acesso em: 18 jun. 2002.

HOLE, Linda. **Usability Evaluation** Disponível em: <http://dec.bournemouth.ac.uk/staff/lhole/hci/hci_3.htm> Acesso em: 10 abr. 2002.

HOLE, Linda. **Users, metaphors and mental models**. Disponível em: <http://dec.bournemouth.ac.uk/staff/lhole/hci.hci_5.html> Acesso em: 17 abr. 2002.

JEFFRIES, Robin et al. **User interface evaluator in the real world: a comparison of four techniques**. Proceedings CHI'91 (ACM – Computer Human-Interaction) (direitos reservados de HP- Hewelett Packard 1990). New Orleans, 1991

JORGE, Joaquim P. **Introdução a IHM.** Disponível em: <http://mega.ist.utl.pt/~ic-ihm/programa/index.html> Acesso em: 18 jun. 2002. (a)

JORGE, Joaquim P. **Métodos de avaliação II**. Disponível em: <http://mega.ist.utl.pt/~ic-ihm/programa/index.html> Acesso em: 18 jul. 2002. (c))

JORGE, Joaquim P. **Técnicas de avaliação I**. Disponível em: <http://mega.ist.utl.pt/~ic-ihm/programa/index.html> Acesso em: 18 jun. 2002. (b)

LAVERY, Darryn, et al. Comparison of evaluation methods using structurede usability problem reports. **Behaviour & Information Technology**, v.16, n. 4/5, 1997, p. 246-266.

MEDEIROS, Marco Aurelio. **ISO 9241: Uma proposta de utilização da norma para avaliação do grau de satisfação de usuários de softwares**. Dissertação de mestrado PPGEPS-UFSC, 1999.

MOÇO, Sueli de Souza. **O uso de cenários como uma técnica de apoio para avaliações ergonômicas de softwares interativos**. Dissertação de mestrado PPGEPS-UFSC, 1996.

MORAES, Anamaria de. **Ergonomia:** Conceitos Disponível em: <http://lableui.vila.bol.com.br/leui.html> Acesso em: 28 ago. 2001.

NIELSEN, Jakob. **Characteristics of usability problems found by heuristic evaluation**. Disponível em: <http://www.useit.com/papers/heuristic/ severityrating.html> Acesso em: 05 ago. 2002.

NIELSEN, Jakob. **Heuristic Evaluation**. Disponível em: <http://www.useit.com/ papers/heuristic> Acesso em: 11 jul. 2002.

NIELSEN, Jakob. **How to Conduct a Heuristic Evaluation**. Disponível em: <http://www.useit.com/papers/heuristic/heuristic_evalutation.html> Acesso em: 11 jul. 2002.

NIELSEN, Jakob. **Summary of usability inspection methods**. Disponível em: <http://www.useit.com/papers/heuristic/inspection_summary.html> Acesso em: 11 jul. 2002.

NIELSEN, Jakob. **Technology Transfer of Heuristic Evaluation and Usability Inspection**. Disponível em: <http://www.useit.com/papers/heuristic/serverityrating. html> Acesso em: 05 ago. 2002.

NIELSEN, Jakob. **Technology transfer of Heuristic evaluation and Usability inspection**. Disponível em: <http://www.useit.com/papers/heuristic/ learning_inspection.html> Acesso em: 05 ago. 2002 (IFIP INTERACT'95, Lillehammer, Norway, June 27, 1995)

NIELSEN, Jakob. **Ten Usability Heuristics**. Disponível em: <http://www.useit.com/papers/heuristic/heuristic_list.html> Acesso em: 15 jul. 2002.

RAMOS, Edla Maria Faust. **Análise ergonômica do sistema hiperNet buscando o aprendizado da cooperação e da autonomia.** Tese de doutorado, UFSC, 1996.

ROCHA, Ana Regina C. **Análise e projeto estruturado de sistemas**. Campus, Rio de Janeiro, 1990.

ROMANI, Luciana A. S. **Proposta de modelo para HCI**. Disponível em: <http://www.cnptia.embrapa.br/~luciana/disciplinas/mod645/tarefa2.html> Acesso em: 21 out. 2001.

SCAPIN, Dominique ; BASTIEN, J. M. Christian. Ergonomic criteria for evaluating the ergonomic quality of interactive systems. **Behaviour & Information Technology**, v.16, n. 4/5, 1997, p. 220-231.

SHNEIDERMAN, B. **Desingning the user interface**: strategies for effective human-computer-interaction. 3ª ed. USA, Addison-Wesley, 1998.

SILVA, C. R. O. **Bases Pedagógicas e Ergonômicas para Concepção e Avaliação de Produtos Educacionais Informatizados.** Disponível em: <http://www.eps.ufsc.br/disserta98/ribeiro/cap4.html> Acesso em: 12 ago. 2002

SILVA, Cinara Nunes ; RAMOS, Liliani Beatriz. **Análise ergonômica de um ambiente de comunicação via Web.** Trabalho de conclusão de graduação, UFSC,

1999, 163 p.

TOGNAZZINI, Bruce. **First Principles**. Disponível em: <http://www.asktog.com/basics/firstPrinciples.html> Acesso em: 11 jul. 2002.

WISNER, Alain. **Por Dentro do trabalho, ergonomia: método e técnica**. Editora FTD, São Paulo, 1987.

ANEXOS

ANEXO 1: EXEMPLO DE TRANSCRIÇÃO DE SESSÃO DE INTERAÇÃO

Códigos utilizados para transcrição das sessões dos ensaios de interação, baseados e adaptados de acordo com os sugeridos por Ramos (1996).

t	Diálogo do usuário e/ou pesquisador
💻	Descrição do estado ou operação realizada pelo computador
[algo]	Leitura efetuada pelo usuário
[janela]	Rolagem da janela para baixo
[janela]	Rolagem da janela para cima
[descrição]	Acionamento do botão "descrição"
[descrição] **op** [opção]	Ativação da "opção" do menu "descrição"
[texto]	Texto digitado pelo usuário
< comentário>	Comentário/observações sobre o estado do *software* ou andamento do trabalho

Sessão 1

Jane

Tarefa: Lançar pedidos

t- *A gente entra no Pedido. Seleciona data, por exemplo, eu vou lançar o pedido para toda a semana, não sei se você já conhecia. A gente pega o pedido, por exemplo, hoje/amanhã para poder lançar para ele todos os dias da semana que vem. Todos os dias que vai ser entregue. Então, qualquer data da semana que vem para poder começar a lançar.*

< Localiza/busca nas 3 linhas disponíveis para localizar onde se encontra o supermercado que deseja lançar.>

< É preciso conhecer em qual linha pertence ou buscar em cada uma linha até encontrar o supermercado desejado.>

eu t- *Você ia lançar na linha de Florianópolis?*

t- *Não. Eu até fui ver, mas é ... da linha Sul, então vou lançar aqui.*

Então pra esse daqui a gente entrega só terça-feira. Então, eu seleciono a data de terça.

<Procura na lista para localizar o produto, clicando em vários itens>

t*Ah, eu ainda não estou bem acostumada com esse sistema porque o outro era totalmente diferente, essa parte aqui...*

eut- *Mas o que tem na tela não segue a sequência que você tem no formulário?*

t- *Nesse aqui... não, meu problema é, meu formulário é assim<mostra o formulário e explica que não teve tempo de colocá-lo no mesmo padrão> Prá mim é complicado porque ele (cliente) manda no formulário que eles têm.*

eut- *No caso o que você fez agora?*

< Digitou um número na quantidade com o sinal + ao lado de outro número.>

t- 20 + 7 no caso, 7 é a troca.

eut- Mas você é quem põe o sinal de mais?

t – hum-hum (sim). Aí, quando não tem produto aqui (na lista da tela) tenho que incluir aqui (mostra o canto inferior esquerdo da tela – " incluir produto" .

eut – Porque, os produtos que você está visualizando ali é o que normalmente a empresa compra? É isso?

t- Isso, eles já ficam automáticos.

eut- Ai você só adiciona os que não constam?

t- Isso, é os que não ficam.

eut- E se você quizesse eliminar algum produto desses?

t- Só zerando. No caso, só se eu zerar ele. Por exemplo, deixar ele em branco, daí...

eut- Da próxima vez ele não aparece?

t- Não. Acho que não pode, acho que ele sempre fica assim. Ai eu acho. Não sei se dá pra escolher assim.

<Segue incluindo valores em silêncio ate preencher a tabela toda.>

t- Chegastes a conhecer o outro sistema que a gente tinha para lançar o pedido que entrava por código?

eut- Vi. Pois é, eu ia lhe perguntar isso. Se para você era mais fácil entrar pelo nome do produto ou pelo código do produto?

t- Pra mim, eu preferia a outra parte, eu até preferia pelo código, porque os códigos tudo, a grande maioria eu já sabia de cor, prá mim era bem mais... Mas colocaram esse prá testar. Porque no caso de um dia eu não poder lançar (o pedido), e alguém lançar pra mim, não ser complicado, porque eles não sabem os códigos, né?!

eut- Mas lá no outro tinha a possibilidade de escolher o nome do produto ou só pelo código?

t- Não. É, mas tinha que procurar lá, selecionando o produto para daí poder incluir.

<Segue incluindo.>

<Pressiona o botão "salvar".>

t- Assim, depois de terminar ele fica tudo verde (*flag*). Assim <mostra outra bandeirinha de outra cor> ele fica com duas cores porque no caso esse aqui não tem pedido. Não foi incluído ainda. Prá gente poder identificar. No caso esse aqui já foi lançado.

eut- Mas nem sempre toda semana você atende todo mundo?

t- Tem semanas que tem mercado que não quer mercadoria. Demora bem mais, antes eu demorava uns 3 minutos prá lançar um pedido desses. Agora... Um pouco também porque eu tô fora de prática. É a segunda vez que eu estou fazendo.

OBS: quando insere um produto, o cursor não fica posicionado no item incluído.

eu†- Você sabe que es'ta no supermercado porque está com a bandeirinha preta? É isso? E dali, você não vai visualizar todo o pedido?

†- Não. Só visualizo depois que terminar tudo.

eu†- Tem alguma forma de conferir? Não?

† – Não. Só visual.

ANEXO 2: EXEMPLO DE RELATÓRIO DE AVALIAÇÃO PRÉVIA ENTREGUE PARA EQUIPE TÉCNICA

Avaliação prévia do Módulo Cadastro

(última atualização: 29/11/2001)

Prioridades:

☹ **problemas de funcionamento (erros/ou necessitam de reparos urgentes)**

😐 **sugestões de melhorias/facilidades de uso**

(depende de definições com os responsáveis do projeto)

☺ **recomendações ergonômicas (versões futuras/ alteração de *lay-out*/melhorias)**

(depende de definições com os responsáveis do projeto)

Item avaliado:

1) CONDOMÍNIO

Tela Dados Cadastrais:

- ☹ no caso de salvar um condomínio com um nome já excluído, a seguinte mensagem é exibida:

O *registro não pode ser exluído ou alterado porque a tabela de cotas-estimativas-semanal-producao inclui registros relacionados.*

- 😐 no campo "*atividade"*, não é visível todo o valor para "processamento de H"

- 😐 ao alterar dados de um condomínio existente, deve-se solicitar confirmação da ação

sugestão:

☺ (homogeneidade) seguir modelo de tela de "clientes categoria notas fiscais" no que diz respeito ao lay-out de botões "incluir xxxx", "excluir xxxx", "salvar" e "cancelar" na parte inferior da tela e uso de botões de avanço/retrocesso.

Tela Produtos Produzidos:

sugestões:

☺ desabilitação dos botões "excluir condomínio" e "incluir condomínio";

☺substituição dos botões "marcar todos" e "desmarcar todos" por uma caixa de seleção posicionada ao lado do rótulo "produto", quando marcada, selecione todos os intens da listagem da caixa abaixo, e quando desmarcada, deixa em branco todas as caixas de seleção dos produtos, para seleção individual, (vale o mesmo para o rótulo "produtos produzidos pelo condomínio")

inclusão de botões "salvar" e "cancelar" na parte inferior direita da tela.

Tela Despesas:

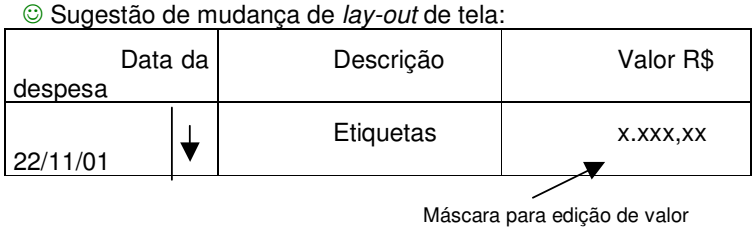
☹ 😐 Usar <TAB> para edição dos campos;

☹ Checar caso de mesma data com mesma descrição de despesa;

☹ Ordenar dados por data;

Inclusão dos botões “incluir despesas”, “excluir despesas”, “salvar” e “cancelar” na parte inferior direita da tela;

☺ Sugestão de mudança de *lay-out* de tela:



2)PRODUTO:

Tela Dados Cadastrais:

😐 Não existem mensagens sobre ações do tipo, confirmação de salvamento, confirmação de inclusão, ou mensagens de erros neste módulo.

😐 Ao salvar produtos produzidos por determinado condomínio não solicita confirmação.

😐 Apresentar dados *default* para itens como “aliquota”, “situação tributária”, “ICMS”, “código do frete” ou colocar caixas de opções para facilitar a entrada de dados e/ou usar exemplos ao lado do rótulo, usando máscaras para edição de dados numéricos, alinhados à direita.

Tela Categoria de Produtos

😐 Estar na mesma cor das outras tabelas de entrada de dados;

☺ Alterar posicionamento dos botões “inserir categoria”, “excluir categoria”,“cancelar” e “salvar” para a parte inferior direita da tela;

* editar diretamente na tabela????

☺ Sugestão: retirar a tela “categoria de produtos” de dentro da opção “produtos” e colocá-la em em uma ficha nova ao lado da opção “produtos”.

Recomendações gerais aos módulos visitados:

☺ Alterar ordem dos botões do canto inferior esquerdo, colocando em primeiro lugar “incluir xxx” e depois “excluir”(caso permaneça esse padrão) ➔ sugerir padrão de botões no campo inferior direito da tela;

😐 Utilizar padrões de simbologia nas mensagens de erros, do tipo:

 Mensagem de caracter informativo;

 Mensagem de alerta (atenção);

 Mensagem de questionamento de ação;

 Mensagem de erro.

☺ Colocação de sons específicos para cada tipo de mensagem, principalmente as críticas.

☺☹ Posicionar o cursor no primeiro campo, para entrada de dados;

☹☺ Salto para o próximo campo usando a tecla <TAB>;

☺ Após o preenchimento de todos os campos, posicionar a ação para o botão *default*, facilitando a conclusão da inclusão usando a tecla <ENTER>;

☺ Padronizar a forma de entrada de dados, através do uso de formulários;

(Obs: em alguns módulos a entrada é feita dentro da própria tabela);

☺☹ Solicitar confirmação para ações do tipo inclusão/exclusão;

☹ Permitir o cancelamento de operações, retornando ao estado anterior;

☹☺ Usar máscara para edições de campos numérios, entradas posicionadas à direita com uso de pontuação e vírgula quando necessário;

☺ Fazer validação de digitação no momento da edição do campo, quando o dado for obrigatório/crítico;

☺ Em botões que levam a outra tela de edição, usar "..."(reticências);

☹ Imprimir condonínios, produtos, trasnportadores, etc., não está contemplado neste módulo.